UNIVERSIDAD DE LA RIOJA

TRABAJO FIN DE ESTUDIOS

Título

Middleware para consumir el API REST del Campus Virtual de la Universidad de la Rioja (Blackboard Learn)

Autor/es

SERGIO CÁRCAMO GARCÍA

Director/es

ÁNGEL LUIS RUBIO GARCÍA

Facultad

Facultad de Ciencia y Tecnología

Titulación

Grado en Ingeniería Informática

Departamento

MATEMÁTICAS Y COMPUTACIÓN

Curso académico

2017-18

Middleware para consumir el API REST del Campus Virtual de la Universidad de la Rioja (Blackboard Learn), de SERGIO CÁRCAMO GARCÍA (publicada por la Universidad de La Rioja) se difunde bajo una Licencia Creative Commons Reconocimiento-NoComercial-SinObraDerivada 3.0 Unported. Permisos que vayan más allá de lo cubierto por esta licencia pueden solicitarse a los titulares del copyright.

publicaciones.unirioja.es
E-mail: publicaciones@unirioja.es

UNIVERSIDAD DE LA RIOJA

Facultad de Ciencia y Tecnología

TRABAJO FIN DE GRADO

Grado en Ingeniería Informática

Middleware para consumir el API REST del Campus Virtual de la Universidad de la Rioja (Blackboard Learn)

Realizado por:

Sergio Cárcamo García

Tutelado por:

Ángel Luis Rubio García

Logroño, Julio,2018

Resumen

El proyecto analiza e implementa un middleware que utilice el API REST de la plataforma del Campus Virtual de la Universidad de la Rioja (Blackboard Learn) sin necesidad de utilizar la plataforma web.

El middleware facilita la modificación de las notas concretas de un alumno que esté cursando una asignatura concreta, pudiendo interactuar con el mismo los profesores que impartan esa asignatura o un administrador del sistema.

Se ha desarrollado una aplicación que presenta los resultados en el entorno de trabajo FileMaker para mostrar el middleware en funcionamiento.

Abstract

The project analyzes and implements a middleware that uses the REST API of the Virtual Campus platform of the University of La Rioja (Blackboard Learn) without using the web platform.

The middleware facilitates the modification of the specific notes of a student who is studying a specific subject, being able to interact with it the professors who teach that subject or a system administrator.

An application has been developed that presents the results in the FileMaker work environment to show the middleware in operation.

Índice

Documento de objetivos del trabajo

Antecedentes

En la Universidad de la Rioja se utiliza la herramienta Blackboard Learn, que es un sistema LMS (Learning Management System, Sistema de Gestión de Aprendizaje) desarrollado por Blackboard Inc. y que permite acceso a contenido de la institución, mediante web, acerca de los profesores, alumnos y los cursos que en ella se imparten. Esta aplicación ha surgido de la evolución del aula universitaria, para una interacción entre profesores y alumnos desde un aula virtual en la red de la universidad.

Este sistema LMS funciona correctamente, aunque con ciertas limitaciones. Adicionalmente, permite el acceso a los contenidos mediante servicios REST que actualmente no están siendo utilizados en el campus virtual de la UR. Por esta razón, se ha propuesto la creación de un middleware que permita consumir esos servicios REST desaprovechados y poder estudiar qué ventajas se obtendrían y qué nuevas limitaciones aportarían los servicios REST, en comparación con el sistema web del aula virtual utilizado actualmente.

Objetivo

El objetivo es desarrollar un middleware que comunique la Plataforma Blackboard Learn con los usuarios, que en este caso podrían ser un administrador o un profesor de la Universidad, para que puedan interactuar más eficientemente con las calificaciones de las asignaturas utilizadas en el Campus Virtual de la UR.

En el proyecto se va a analizar cuál es la solución más óptima tanto por seguridad como por sencillez, para ello se utilizarán, entre otras, que dependan de ellas, las siguientes tecnologías, Blackboard Learn, servicios REST y FileMaker.

Se va a utilizar FileMaker porque es un requisito del tutor de empresa, ya que facilita el acceso a datos mediante sencillos formularios a través de una red compartida.

Los datos se obtendrán desde Blackboard Learn y mediante el middleware podrán procesarse desde FileMaker mediante unos sencillos formularios, teniendo el usuario la posibilidad de modificar los datos y exportarlos para tenerlos almacenados en otro formato.

De esta manera se desarrollará una aplicación fácilmente usable desde FileMaker para que en un futuro pudieran desarrollarse otras mejoras para el campus virtual de la Universidad de la Rioja.

Objetivos relacionados con el proyecto:

- Instalación de una instancia virtualizada de la plataforma Blackboard Learn para la simulación de las pruebas correspondientes
- Implementación de un cliente consumidor de REST
- Conseguir la comunicación entre el cliente consumidor de REST y FileMaker
- Creación de formularios amigables para los profesores y el administrador, en FileMaker

Estructura

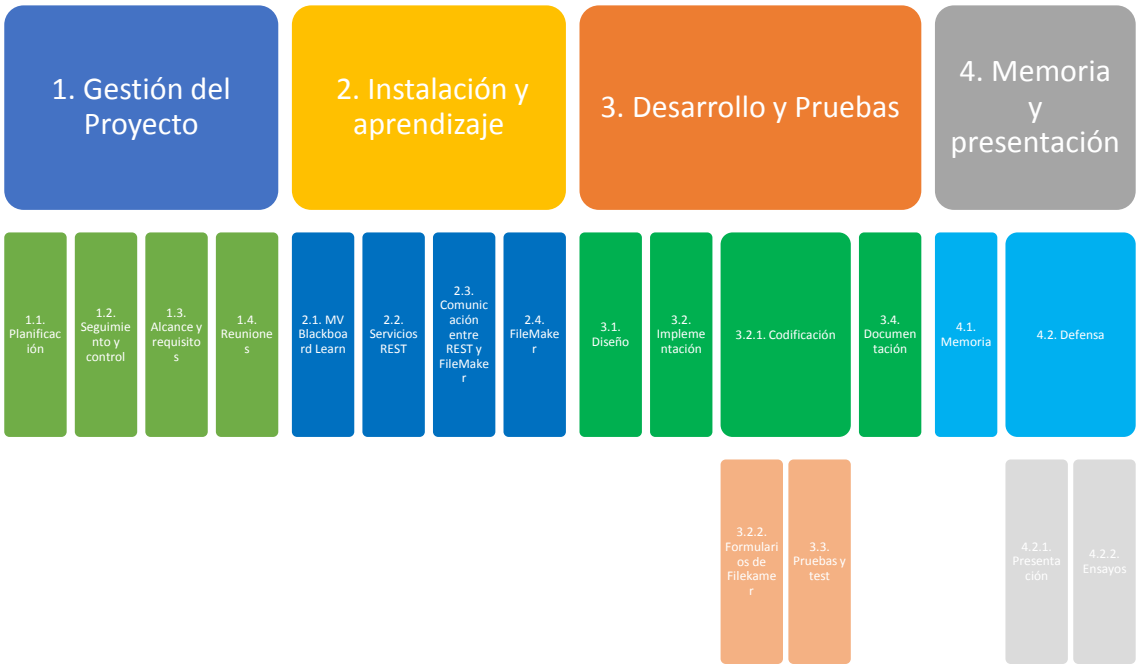


Figura 1. Descomposición de tareas del proyecto

A continuación, se van a describir las tareas:

1. Gestión del Proyecto

1.1. Planificación

Definición de la tareas, cronograma e hitos del proyecto, dicho contenido será incluido en la memoria.

1.2. Seguimiento y control

Anotaciones del seguimiento del proyecto comparándolo con lo planificado. Aquí se incluirán posibles rectificaciones en la planificación cuando sean necesarias. Al final de la memoria se incluirá un informe detallado sobre cómo ha sido la evolución del proyecto.

1.3. Alcance y requisitos

Definición de qué se va a hacer y qué es lo que no se puede hacer.

1.4. Reuniones

Reuniones con los tutores para tener un seguimiento y resolver dudas. Se estima una media de una reunión cada dos semanas con una duración aproximada de una hora.

2. Instalación y aprendizaje

2.1. MV Blackboard Learn

Instalación de la máquina virtual. Backup de la misma para poder transportarla a otras máquinas. Comprensión del entorno. Creación de datos de prueba que serán usados más adelante, tales como alumnos, asignaturas y cursos. Lectura de documentación relacionada.

2.2. Servicios REST

Lectura y comprensión de documentación de los servicios REST. Elección de un cliente consumidor de REST.

2.3. Comunicación entre REST y FileMaker

Lectura y comprensión de documentación de la comunicación de servicios REST con FileMaker. Instalación y configuración de la comunicación necesaria para acceder a los datos de forma bidireccional desde los servicios REST hacia FileMaker.

2.4. FileMaker

Diseño y creación de formularios para que los profesores accedan a los datos de un determinado curso y puedan poner notas a los distintos alumnos. En el caso de ser un administrador, deberían tener un acceso total a los datos de profesores, alumnos, cursos y notas de los distintos alumnos.

3. Desarrollo y Pruebas

3.1. Diseño

Se definirán las distintas partes que compondrán la aplicación y se definirá cómo lo van a hacer.

3.2. Codificación

Se codifica en los lenguajes requeridos para crear la aplicación.

3.2.1. Formularios de FileMaker

Se diseñan y se crean los formularios que usarán los profesores para la gestión de las notas. También se diseñarán y crearán los formularios que usará un administrador para realizar las tareas de gestión.

3.2.2. Pruebas y Test

Se realizan pruebas y test de conexiones, validez de datos, seguridad, para comprobar que todo funciona correctamente y como se espera que lo haga.

3.3. Documentación

Documentación para el código generado. Además de anotaciones sobre errores encontrados, posibles soluciones encontradas, mejoras que podrían implementarse en un futuro y cualquier otra información útil para terceros que necesiten documentarse sobre la aplicación.

4. Memoria y presentación

4.1. Memoria

Redacción de la memoria, así como su corrección y ensamblado ante cada parte que la compone.

4.2. Defensa

4.2.1. Presentación

Diseño, preparación y maquetado de unas diapositivas para acompañar la presentación del proyecto terminado.

4.2.2. Ensayos

Preparación del guion que se usará junto con las diapositivas y aprendizaje del mismo para hacer más llevadera la presentación.

Planificación

Para llevar cabo las distintas tareas se ha elegido la metodología ágil SCRUM, dividiendo el trabajo en varios Sprints numerados del 1 al 9:

S1. Configuración de la Plataforma Blackboard Learn: 30 horas
S2. Estudio del panel de administración Blackboard Learn: 30 horas
S3. Desarrollo Servicios REST: 30 horas
S4. Implementación del Middleware: 30 horas
S5. Diseño de los formularios en FileMaker: 30 horas
S6. Implementación de la funcionalidad en FileMaker: 30 horas
S7. Memoria: 25 horas
S8. Presentación: 25 horas
S9. Ensayos: 3 horas

Los Sprints 1 al 6 tienen asociada una etapa de pruebas y de reuniones para detectar y corregir errores. Se han elegido tiempos de 30 horas para que tengan una duración similar, no se ha tenido en cuenta el grado de dificultad de los mismos.

Los Sprints 7 al 9 están más centrados en generar la documentación del proyecto y la presentación ante el tribunal, también tienen asociada una reunión para terminar de prepararlos correctamente. Se han elegido tiempos de duración similar, de 25 horas, para las tareas de gestión y de 3 horas para los ensayos.

Además de las horas mencionadas antes hay que añadir las siguientes tareas de Gestión:

G1. Planificación, documento de objetivos del trabajo, corrección de la memoria del proyecto y documentación: 40 horas
G2. Reuniones: 13 horas, aproximadamente de 1 hora de duración cada una
G3. Pruebas: 14 horas

Tarea	**Horas**
Sprints	233
Planificación, Memoria y documentación	40
Reuniones	13
Pruebas	14
Total	**300**

Figura 2: relación de horas

N.º de Tarea	Nombre de la Tarea	Duración en horas	Inicio	Fin	Predecesor
G2.1	Reunión con el Tutor de la Empresa	1	Jueves 01/02/18	Jueves 01/02/18	
S1	Configuración de la Plataforma Blackboard Learn	30	Jueves 01/02/18	Lunes 12/02/18	
G2.2	Reunión con el Tutor de la Universidad	1	Lunes 05/02/18	Lunes 05/02/18	
G1.1	Planificación	20	Lunes 05/02/18	Viernes 09/02/18	
G1.2	Documento de objetivos del trabajo	20	Lunes 05/02/18	Lunes 12/02/18	
G2.3	Reunión con el Tutor de la Universidad	1	Lunes 12/02/18	Lunes 12/02/18	G1.2
G2.4	Reunión con el Tutor de la Empresa	1	Lunes 12/02/18	Lunes 12/02/18	S1
S2	Estudio del panel de administración Blackboard Learn	30	Martes 13/02/18	Jueves 22/02/18	S1
S3	Desarrollo Servicios REST	30	Lunes 26/02/18	Jueves 08/03/18	S2
G2.5	Reunión con el Tutor de la Universidad	1	Viernes 09/03/18	Viernes 09/03/18	S3
G2.6	Reunión con el Tutor de la Empresa	1	Viernes 09/03/18	Viernes 09/03/18	S3
S4	Implementación del Middleware	30	Martes 13/03/18	Viernes 23/03/18	S3
G3.1	Pruebas de los Servicios REST	7	Lunes 26/03/18	Martes 27/03/18	S3
G2.7	Reunión con el Tutor de la Universidad	1	Lunes 28/03/18	Lunes 28/03/18	S4; G3.1
G2.8	Reunión con el Tutor de la Empresa	1	Lunes 28/03/18	Lunes 28/03/18	S4; G3.1
S5	Diseño de los formularios en FileMaker	30	Jueves 29/03/18	Martes 10/04/18	S4; G3.1
G2.9	Reunión con el Tutor de la Universidad	1	Lunes 11/04/18	Lunes 11/04/18	S5
G2.10	Reunión con el Tutor de la Empresa	1	Lunes 11/04/18	Lunes 11/04/18	S5
S6	Implementación de la funcionalidad en FileMaker	30	Jueves 12/04/18	Lunes 23/04/18	S5
G3.2	Pruebas de la funcionalidad en FileMaker	7	Martes 24/04/18	Jueves 26/04/18	S6
G2.11	Reunión con el Tutor de la Universidad	1	Viernes 27/04/18	Viernes 27/04/18	S6; G3.2
G2.12	Reunión con el Tutor de la Empresa	1	Viernes 27/04/18	Viernes 27/04/18	S6; G3.2
S7	Memoria	25	Lunes 30/04/18	Martes 08/05/18	S1; G1.1; G1.2; S2; S3; S4; G3.1; S5; S6; G3.2
S8	Presentación	25	Miércoles 09/05/18	Viernes 18/05/18	S7
S9	Ensayos	3	Lunes 21/05/18	Lunes 21/05/18	S8
G2.13	Reunión con el Tutor de la Universidad	1	Viernes 08/06/18	Viernes 08/06/18	S7; S8; S9
	Total	300			

Figura 3: Hitos y fechas asignadas las tareas

Diseño de la Aplicación

Arquitectura

Se va a utilizar un diseño en 3 capas:

1. **Persistencia:** mediante un servidor Blackboard Learn, se gestiona todo lo relativo a la base de datos y a la creación, edición y borrado de datos de ésta. También gestiona el acceso de los usuarios y da permisos a los desarrolladores (mediante tokens) para poder tener acceso mediante las aplicaciones que desarrollen.

2. **Negocio:** mediante el API de servicios REST, se gestiona la lógica de la aplicación. Aquí es donde se dice qué hacer con los datos, qué usuarios podrán manejar los datos y cómo podrán manejarlos. Por ejemplo, para consultar la nota de un alumno inscrito en un curso, la aplicación deberá gestionar que se tienen permisos para hacer la consulta, comprobar que se puede hacer la consulta y devolver o los datos o un mensaje de que la consulta ha fallado. Estará conectada con la capa de persistencia para poder realizar sus funciones a partir de los datos que le sean suministrados.

3. **Presentación:** mediante el programa FileMaker se creará la interfaz del usuario. Su única función es pasarle las acciones que realice el usuario a la capa de negocio, utilizando el Api REST mediante comandos CURL. La aplicación final se centrará en esta capa y usará los datos recibidos como Json que previamente recibirá de la capa de negocio, que en algunos casos concretos podrá devolver parcialmente esos datos a la capa de negocio en forma de Json. Adicionalmente, se podrá obtener una exportación de los datos a un fichero en uno de los formatos que FileMaker soporta.

A continuación, se expone un diagrama del diseño de 3 capas de la aplicación.

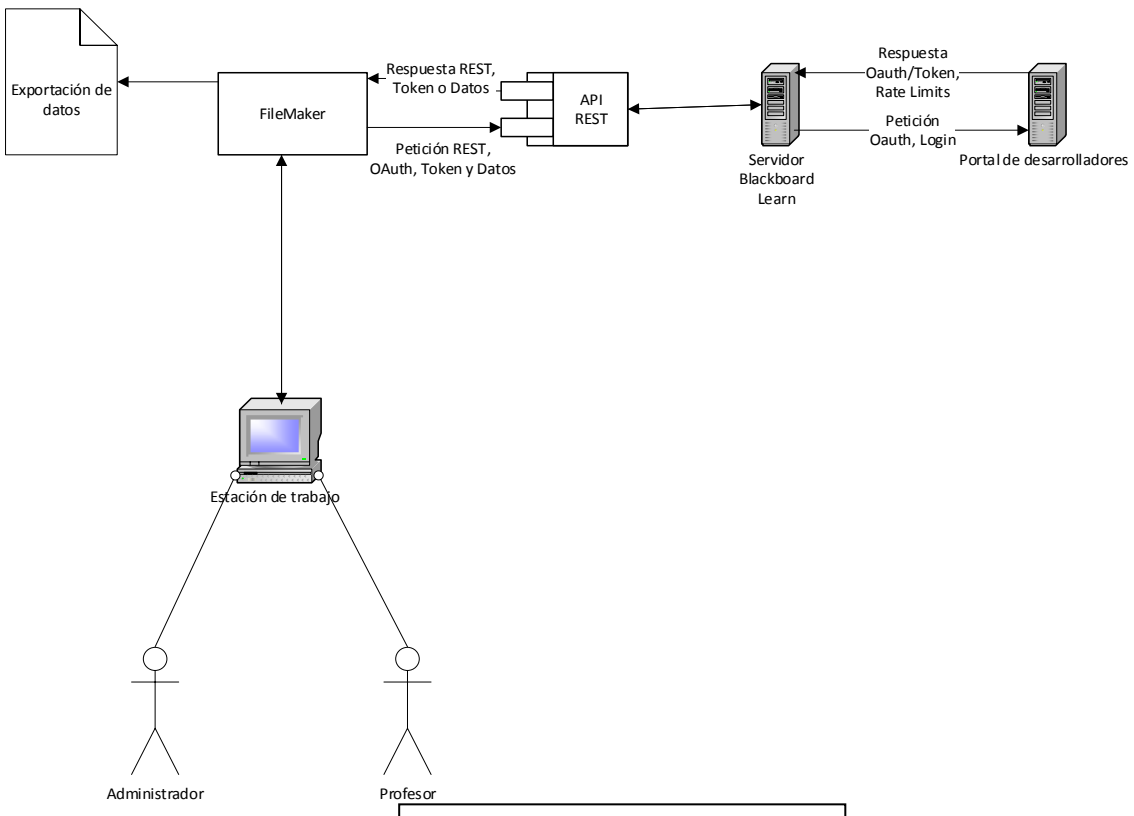


Diagrama 1. Aplicación de 3 capas

Casos de uso

A continuación, se expone el diagrama de uso de la aplicación.

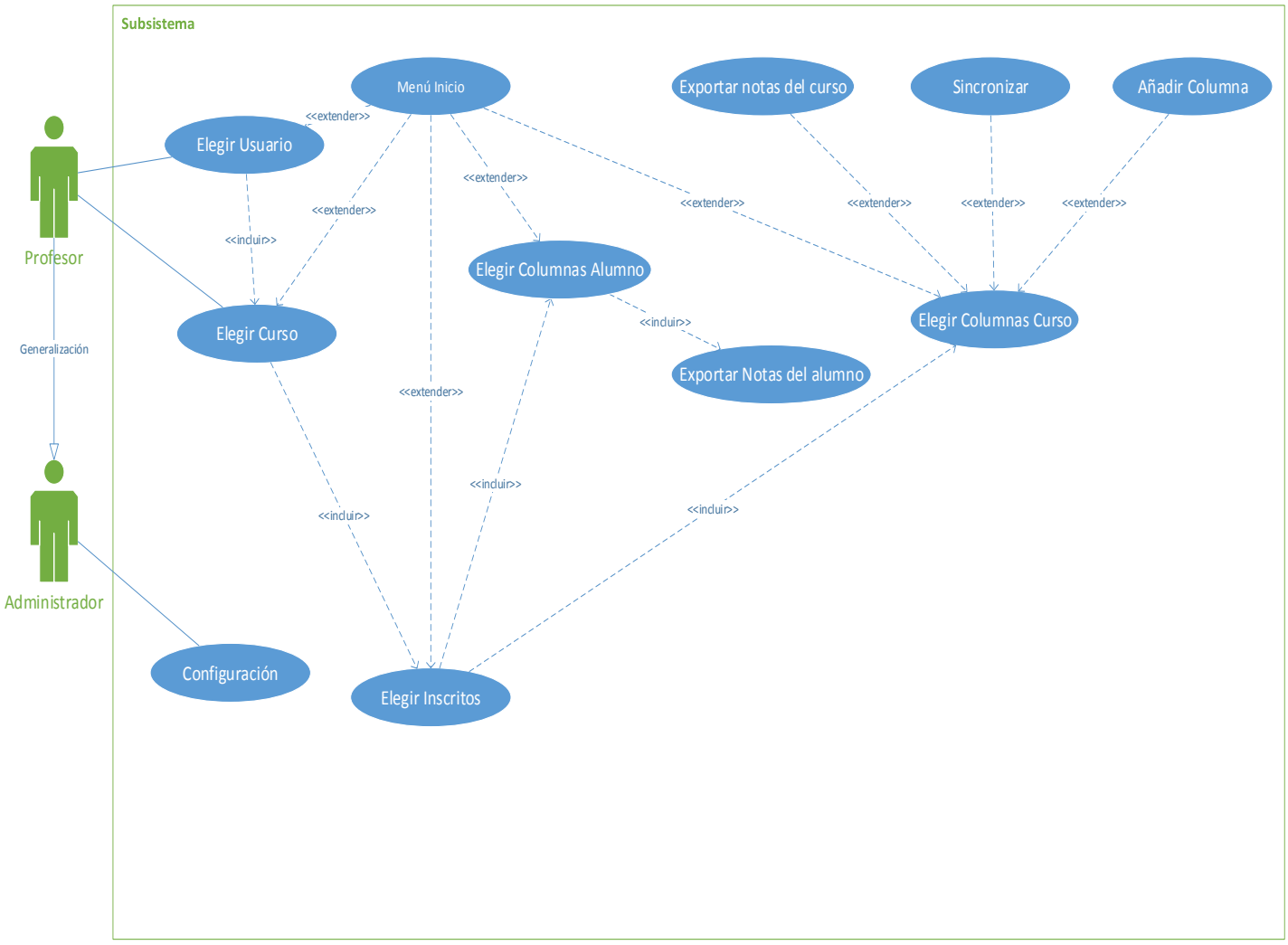


Diagrama de casos de Uso 1. Subsistema de la aplicación

Caso de Uso	Configuración
Descripción	Un usuario Administrador modifica la configuración en el sistema
Actor	• Administrador
Precondición	• La conexión de red está activa • Blackboard Learn está correctamente configurado • El usuario Administrador existía previamente en el sistema
Postcondición	• La configuración introducida genera un token válido

Secuencia Normal	• Se busca una configuración previa perteneciente al usuario
	  ○ Si la encuentra, la muestra
	  ○ Si no la encuentra, se muestra la configuración por defecto del usuario Administrator
	• El usuario puede elegir un botón:
	  ○ Aceptar, se confirma la configuración
	  ▪ Se comprueba la validez de la configuración:
	  • Se comprueba si url, key o secret tienen un valor distinto de vacío
	  ○ url, key o secret tienen un valor distinto de vacío:

	▪ Se envía una petición de token para comprobar la conexión ▪ La aplicación devuelve el token en forma de JSON ▪ Se consignan los datos en la tabla ▪ Se vuelve al menú inicial o Cancelar, se vuelve al menú inicial
Excepciones	• Se recibe un código de error o La conexión de red no está activa o Se muestra un mensaje de error indicándolo o Se vuelve al menú inicial • El campo url, key o secret tienen un valor vacío o Se muestra un mensaje advirtiéndoselo al usuario o Se restaura el valor anterior • El campo url no empieza por 'http://'y termina con 9876 o empieza por 'https://' y termina con 9877 o El textBox cambia a un fondo rojo para indicar al usuario que no es una url válida

Caso de Uso	Menú inicio
Descripción	El usuario regresa al menú inicial
Actor	• Administrador • Profesor
Precondición	• La conexión de red está activa • El usuario poseía una configuración válida con un token de conexión no caducado
Postcondición	• El token del usuario pasa al estado de caducado
Secuencia Normal	• El usuario se desconecta • Se envía una petición de token para comprobar la conexión o Modifica las variables globales que permiten la petición de un nuevo token o Se borran los datos de las tablas locales • La aplicación muestra el menú inicial
Excepciones	• El token del usuario ya estaba caducado o no coincide o La aplicación muestra el menú inicial • La conexión de red no está activa o Se muestra un mensaje de error indicando un fallo de conexión

Caso de Uso	Elegir Usuario
Descripción	El usuario elige el usuario activo
Actor	• Administrador • Profesor
Precondición	• El usuario debe existir
Postcondición	• Se recibe un token válido
Secuencia Normal	• El usuario elige un filtro para buscar por: o ID del alumno o Nombre de usuario o Nombre (por defecto) o Apellidos o Correo Electrónico • El usuario elige un filtro para contiene: o Que contiene (por defecto) o Igual a o Comienza por • El usuario escribe en el textBox el valor que quiere buscar • El usuario pulsa el botón Buscar • Se realiza una petición Get para pedir los datos de esos usuarios • Se actualiza el portal con los datos de los usuarios encontrados que cumplen los filtros

	• El usuario puede pulsar el botón:
	  ○ Curso dentro de una fila concreta, para ir al menú Elegir Curso
	  ○ Menú inicial para volver al menú inicial

Excepciones	• El usuario no existía
	  ○ La lista de usuarios está vacía
	• La petición Get devuelve un código de error
	  ○ Se muestra un mensaje indicando el tipo de error
	• La conexión de red no está activa
	  ○ Se muestra un mensaje de error indicando un fallo de conexión
	  ○ Se regresa al menú inicial

Caso de Uso	Elegir Curso
Descripción	El usuario elige el curso activo
Actor	• Administrador • Profesor
Precondición	• El curso debe existir
Postcondición	• El curso es correcto

Secuencia Normal	• El usuario elige un filtro de fecha:
	  ○ Después
	  ○ Antes (por defecto)
	• Se comprueba si el filtro de fecha era vacío, en ese caso se muestra la fecha y hora actual. El usuario puede cambiarlo al pulsar sobre el calendario desplegable
	• El usuario elige un filtro para buscar por:
	  ○ ID del curso
	  ○ Nombre del curso (por defecto)
	  ○ Descripción
	  ○ Profesor
	• El usuario elige un filtro para contiene:
	  ○ Que contiene (por defecto)
	  ○ Igual a
	  ○ Comienza por
	• El usuario escribe en el textBox el valor que quiere buscar
	• El usuario pulsa el botón Buscar
	• Se realiza una petición Get para pedir los datos de esos usuarios
	• Se comprueba si se filtraba por Profesor y si tiene el rol de SystemAdmin
	  ○ Si el rol era SystemAdmin, se leen los datos de todos los cursos y después se muestran sólo los que cumplen los filtros
	  ○ Si el rol no era SystemAdmin se muestran sólo los datos de los cursos que cumplen los filtros
	• El usuario puede pulsar el botón:
	  ○ Alumnos inscritos dentro de una fila concreta, para ir al menú Elegir Inscritos
	  ○ Menú inicial para volver al menú inicial

Excepciones	• El curso no existía
	  ○ La lista de cursos está vacía
	• La petición Get devuelve un código de error
	  ○ Se muestra un mensaje indicando el tipo de error
	• La conexión de red no está activa
	  ○ Se muestra un mensaje de error indicando un fallo de conexión
	  ○ Se regresa al menú inicial

Caso de Uso	Elegir Inscritos
Descripción	El usuario elige un alumno inscrito en un curso
Actor	• Administrador • Profesor

Precondición	• El curso debe existir • El curso debe estar activo • Debe haber al menos un usuario con el rol Instructor
Postcondición	

Secuencia Normal	• Se realiza una petición Get para pedir los datos de los alumnos inscritos en ese curso
	• Se actualiza el portal con los datos del alumno
	• El usuario puede elegir pulsar un botón:
	  ○ Columnas de notas del curso que permite ir a la pantalla Elegir columnas Curso
	  ○ Volver que permite regresar a la pantalla Elegir Curso
	  ○ Menú inicial para volver al menú inicial

Excepciones	• El curso no existía o no se encuentran alumnos o profesores
	  ○ La lista de alumnos inscritos está vacía
	• La petición Get devuelve un código de error
	  ○ Se muestra un mensaje indicando el tipo de error
	• La conexión de red no está activa
	  ○ Se muestra un mensaje de error indicando un fallo de conexión
	  ○ Se regresa al menú inicial

Caso de Uso	Elegir Columnas Curso
Descripción	El usuario realiza una de las operaciones de modificación (creación, sincronización) de las columnas de notas de un alumno inscrito en un curso concreto
Actor	• Administrador • Profesor
Precondición	• El curso debe existir • El alumno debe existir y estar inscrito en el curso • La columna asociada a la nota del alumno inscrito en un curso concreto debe existir
Postcondición	• La nota del alumno inscrito en un curso concreto es correcta

Secuencia Normal	• Se realiza una petición Get para pedir los datos de las columnas de notas del alumno
	• Se actualiza el portal con todas las columnas de notas encontradas
	• El usuario puede elegir pulsar un botón:
	  ○ Sincronizar, para actualizar los cambios en el nombre y la descripción de la columna en Blackboard correspondientes a esa fila
	  ○ Añadir columna
	  ○ Exportar notas del curso
	  ○ Volver que permite regresar a la pantalla Elegir Inscritos
	  ○ Menú inicial para volver al menú inicial

Excepciones	• El nombre de la columna estaba vacío al crear la columna
	  ○ Se muestra un mensaje indicándolo
	• Se trata de modificar el nombre de la columna una vez creada
	  ○ Se muestra un mensaje indicando que no se puede
	  ○ Se restaura el estado anterior
	• La descripción de la columna estaba vacío
	  ○ Se muestra un mensaje indicándolo
	• La petición Get devuelve un código de error
	  ○ Se muestra un mensaje indicando el tipo de error
	  ○ Se restaura el valor anterior
	• La conexión de red no está activa
	  ○ Se muestra un mensaje de error indicando un fallo de conexión
	  ○ Se regresa al menú inicial

Caso de Uso	Elegir Columnas Alumno
Descripción	El usuario realiza la operación de modificación (sincronización, eliminación) de las columnas de notas de un alumno inscrito en un curso concreto
Actor	• Administrador • Profesor
Precondición	• El curso debe existir • El alumno debe existir y estar inscrito en el curso • La columna asociada a la nota del alumno inscrito en un curso concreto debe existir
Postcondición	• La nota del alumno inscrito en un curso concreto es correcta

Secuencia Normal	• Se realiza una petición Get para pedir los datos de las columnas de notas del alumno
	• Se actualiza el portal con todas las columnas de notas encontradas
	• El usuario puede elegir pulsar un botón:
	  ○ Sincronizar, para actualizar los cambios en las notas de la columna en Blackboard correspondientes a esa fila
	  ○ Exportar notas del alumno
	  ○ Volver que permite regresar a la pantalla Elegir Inscritos
	  ○ Menú inicial para volver al menú inicial

Excepciones	• La nota no existía
	  ○ Al mostrar la información se muestra una cadena vacía
	• La nota introducida no está en el rango permitido (número decimal entre 0 y 10)
	  ○ Si se introduce una letra en vez de un número, la letra se elimina
	  ○ Si se introduce una coma para la separación decimal, se sustituye por un punto
	  ○ Al escribir un número entre 0 y 9, se añade punto al final automáticamente para que el usuario pueda añadir hasta 4 decimales
	  ○ Si el valor es negativo o menor que 0, se sustituye por 0
	  ○ Si el valor es superior a 10, se sustituye por 10
	• La petición Get devuelve un código de error
	  ○ Se muestra un mensaje indicando el tipo de error
	  ○ Se restaura el valor anterior
	• La conexión de red no está activa
	  ○ Se muestra un mensaje de error indicando un fallo de conexión
	  ○ Se regresa al menú inicial

Caso de Uso	Exportar Notas del curso
Descripción	El usuario realiza la exportación de todas las columnas de notas de un curso, agrupadas por alumnos inscritos
Actor	• Administrador • Profesor
Precondición	• El curso debe existir • Debe existir al menos un alumno y estar inscrito en el curso • Las columnas asociadas al alumno inscrito deben existir
Postcondición	• Se guarda un fichero con los datos exportados en el formato elegido por el usuario
Secuencia Normal	• Se realiza una petición Get para pedir los datos de las columnas de notas de todos los alumnos de ese curso • Se actualiza el portal con todas las columnas de notas encontradas • Se guarda la información con el formato CSV (primera fila son los nombres de las columnas, separación por comas y separación de filas con salto de línea) en la ruta de documentos del usuario con el nombre “notas_alumnos_<nombre del curso>.csv” ○ Se regresa a la pantalla de Elegir Inscritos

Excepciones	• La petición Get devuelve un código de error
	  ○ Se muestra un mensaje indicando el tipo de error
	• La conexión de red no está activa
	  ○ Se muestra un mensaje de error indicando un fallo de conexión
	  ○ Se regresa al menú inicial

Caso de Uso	Exportar Notas del Alumno
Descripción	El usuario realiza la exportación de todas las columnas de notas de un alumno en curso concreto
Actor	• Administrador • Profesor
Precondición	• El curso debe existir • El alumno debe existir y estar inscrito en el curso • Las columnas asociadas al alumno inscrito deben existir
Postcondición	• Se guarda un fichero con los datos exportados en el formato elegido por el usuario
Secuencia Normal	• Se realiza una petición Get para pedir los datos de las columnas de notas del alumno de ese curso • Se actualiza el portal con todas las columnas de notas encontradas • Se guarda la información con el formato CSV (primera fila son los nombres de las columnas, separación por comas y separación de filas con salto de línea) en la ruta de documentos del usuario con el nombre “nota_<nombre del alumno>.csv” ○ Se regresa a la pantalla de Elegir Columnas

Excepciones	• La petición Get devuelve un código de error
	  ○ Se muestra un mensaje indicando el tipo de error
	• La conexión de red no está activa
	  ○ Se muestra un mensaje de error indicando un fallo de conexión
	  ○ Se regresa al menú inicial

Caso de Uso	Añadir columna
Descripción	El usuario crea una nueva columna de notas dentro del portal
Actor	• Administrador • Profesor
Precondición	• El curso debe existir • El alumno debe existir y estar inscrito en el curso • Las columnas asociadas al alumno inscrito deben existir
Postcondición	• La columna recién creada es válida
Secuencia Normal	• Se añade una columna de notas nueva al portal • La aplicación se queda a la espera de las modificaciones

Excepciones	• Se pulsa Volver o Cerrar sesión
	  ○ Se muestra un mensaje al usuario recordándole que no ha sincronizado:
	  ▪ Si se acepta sincronizar, se validan
	  ○ Si se cancela, no se hace nada

Caso de Uso	Sincronizar
Descripción	El usuario actualiza en Blackboard los cambios para una fila de columnas de notas dentro del portal
Actor	• Administrador

	• Profesor
Precondición	• El curso debe existir • El alumno debe existir y estar inscrito en el curso • Las columnas asociadas al alumno inscrito deben existir
Postcondición	• La columna sincronizada hace firmes los cambios en Blackboard

Secuencia Normal	• Se confirma con el usuario que se quiere sincronizar
	  ○ Se acepta, y se continua con la sincronización
	  ▪ Se comprueba si era una columna nueva en el portal
	  • En caso de serlo se realiza una llamada Post para crearla en Blackboard
	  • En caso de ya existir se hace una llamada Patch para actualizar los cambios en Blackboard
	  ○ Se cancela, no se hace nada
	• Se restaura, se devuelve la fila a su ultimo estado

Excepciones	• La columna era calculada
	  ○ Se muestra un mensaje indicándolo
	• Ha habido cambios al cambiar a True la columna externa
	  ○ Se comprueba cual era la que estaba activa y si se pone a False
	  ○ Se hace una llamada Patch para actualizar los cambios en Blackboard
	  ○ Se comprueba si recibimos un código de error
	  ▪ Recibimos un código de error
	  • Se muestra un mensaje según el tipo de error
	  • Se devuelve el portal a su estado anterior
	  ▪ No recibimos un código de error
	  ○ Se confirman los cambios en el portal

Diagrama de Actividad

A continuación, se exponen los diagramas de actividad de la aplicación. Representan el comportamiento dinámico del sistema haciendo hincapié en los distintos formularios con los que el usuario puede interactuar.

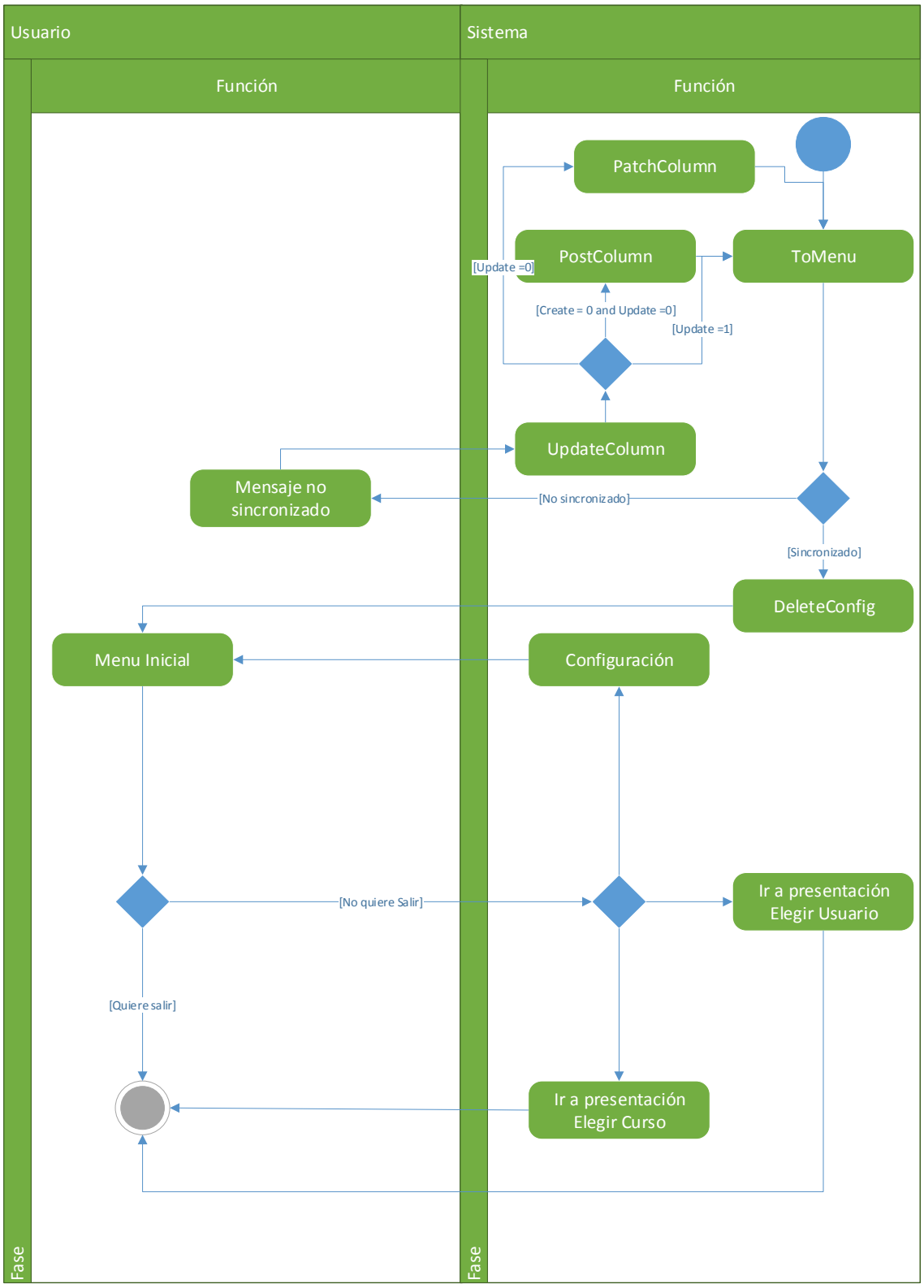


Diagrama de actividad 1. Menú inicial

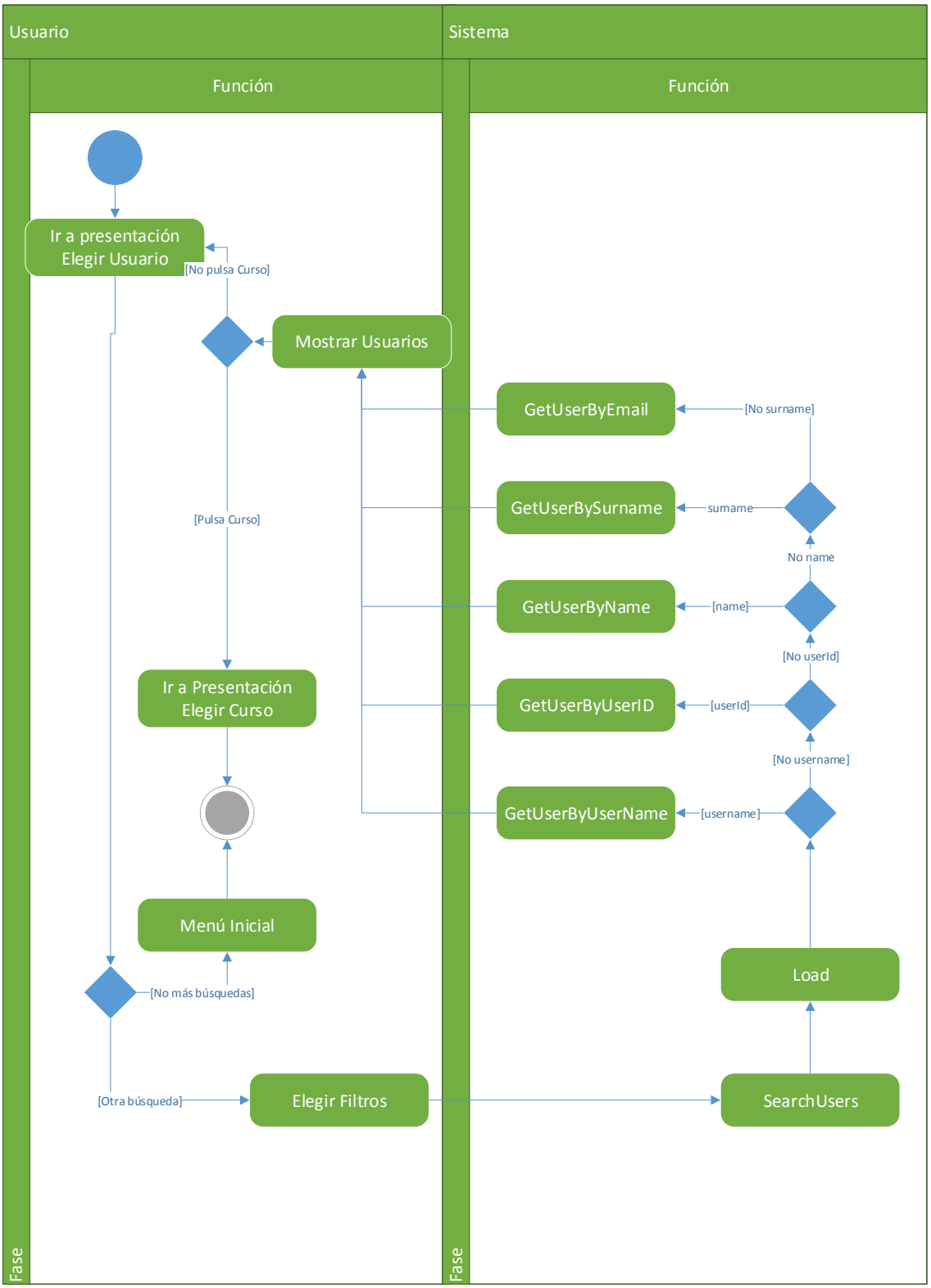


Diagrama de actividad 2. Elegir Usuario

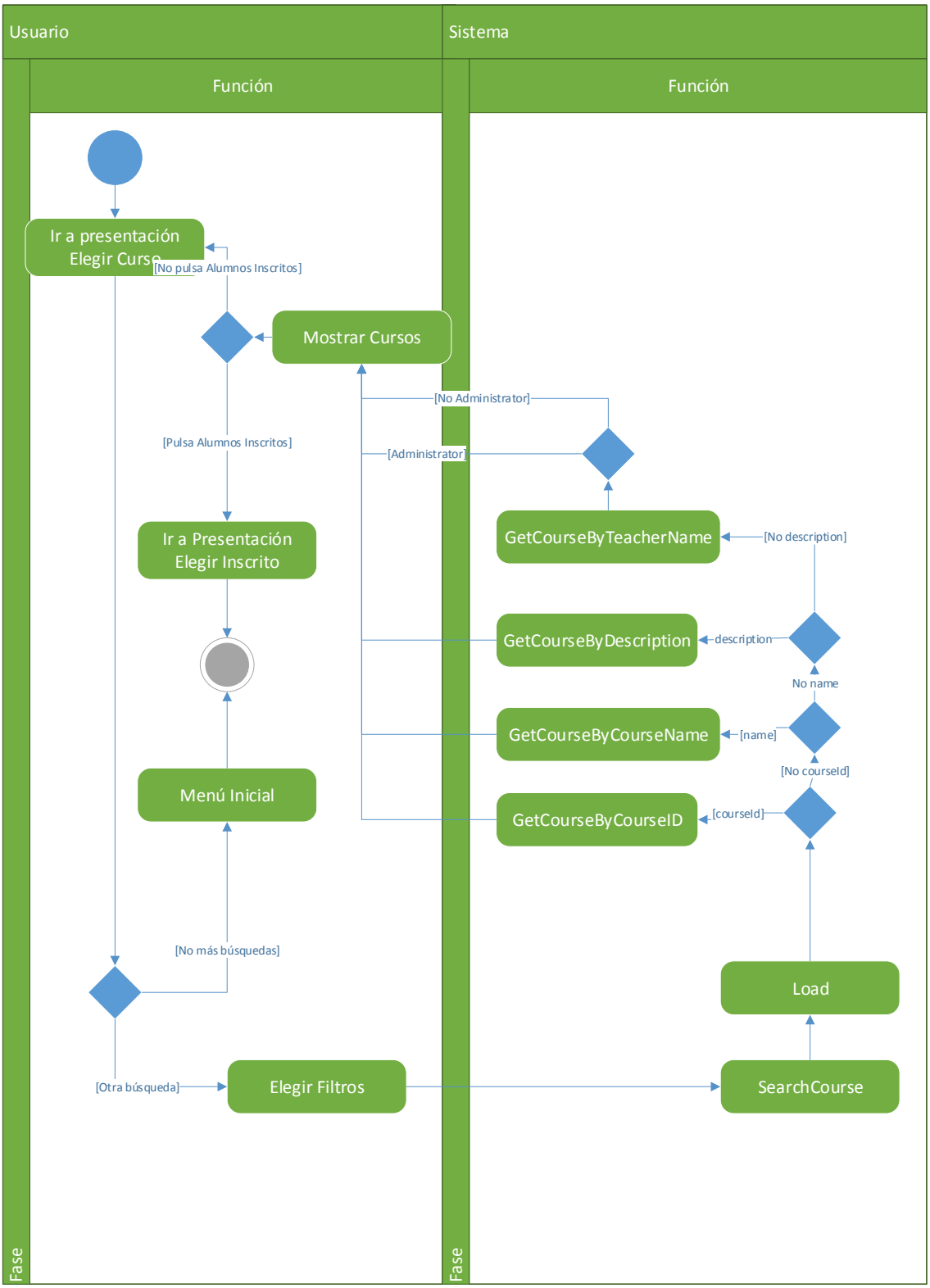


Diagrama de actividad 3. Elegir Curso

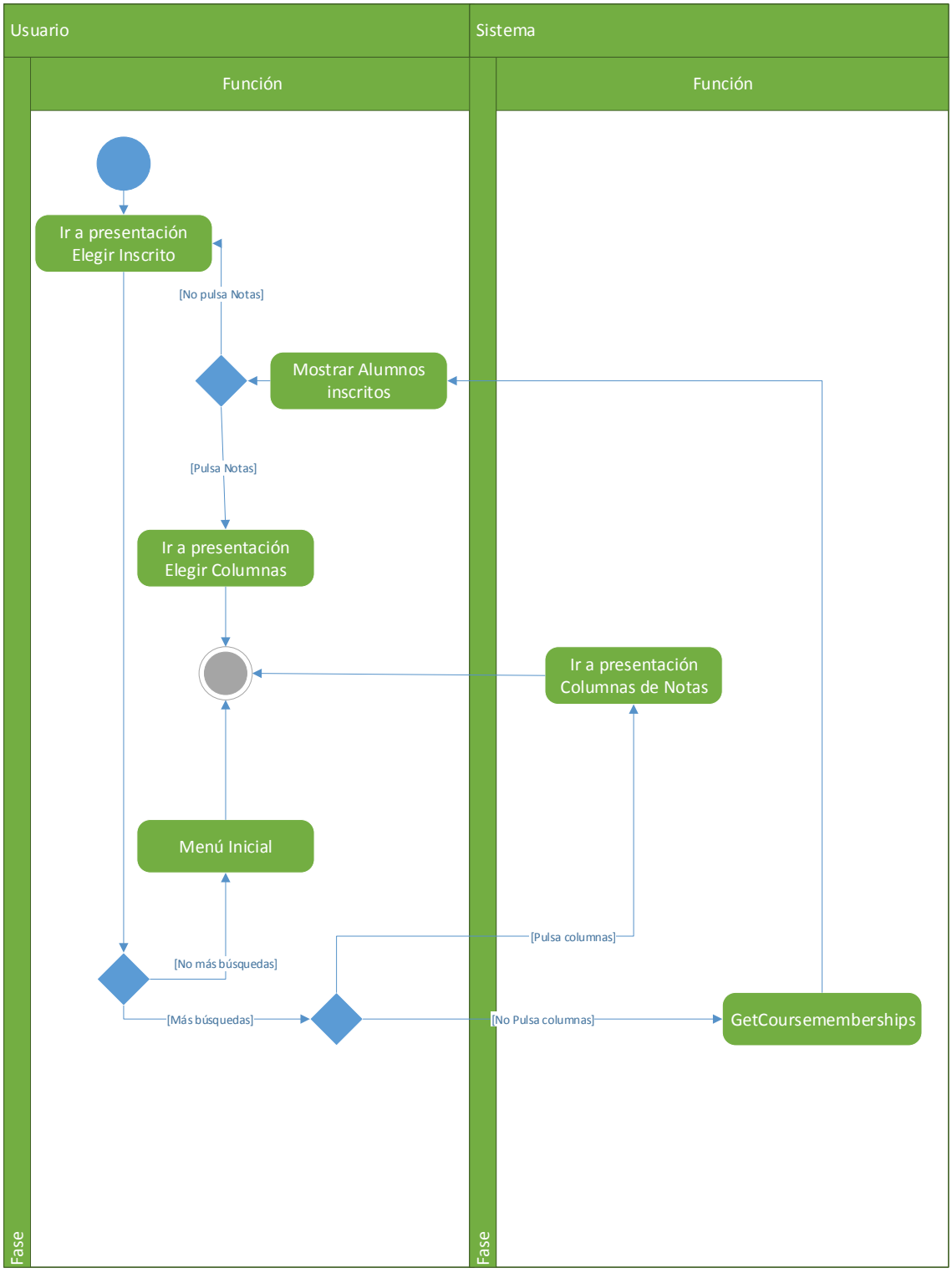


Diagrama de actividad 4. Elegir Inscritos

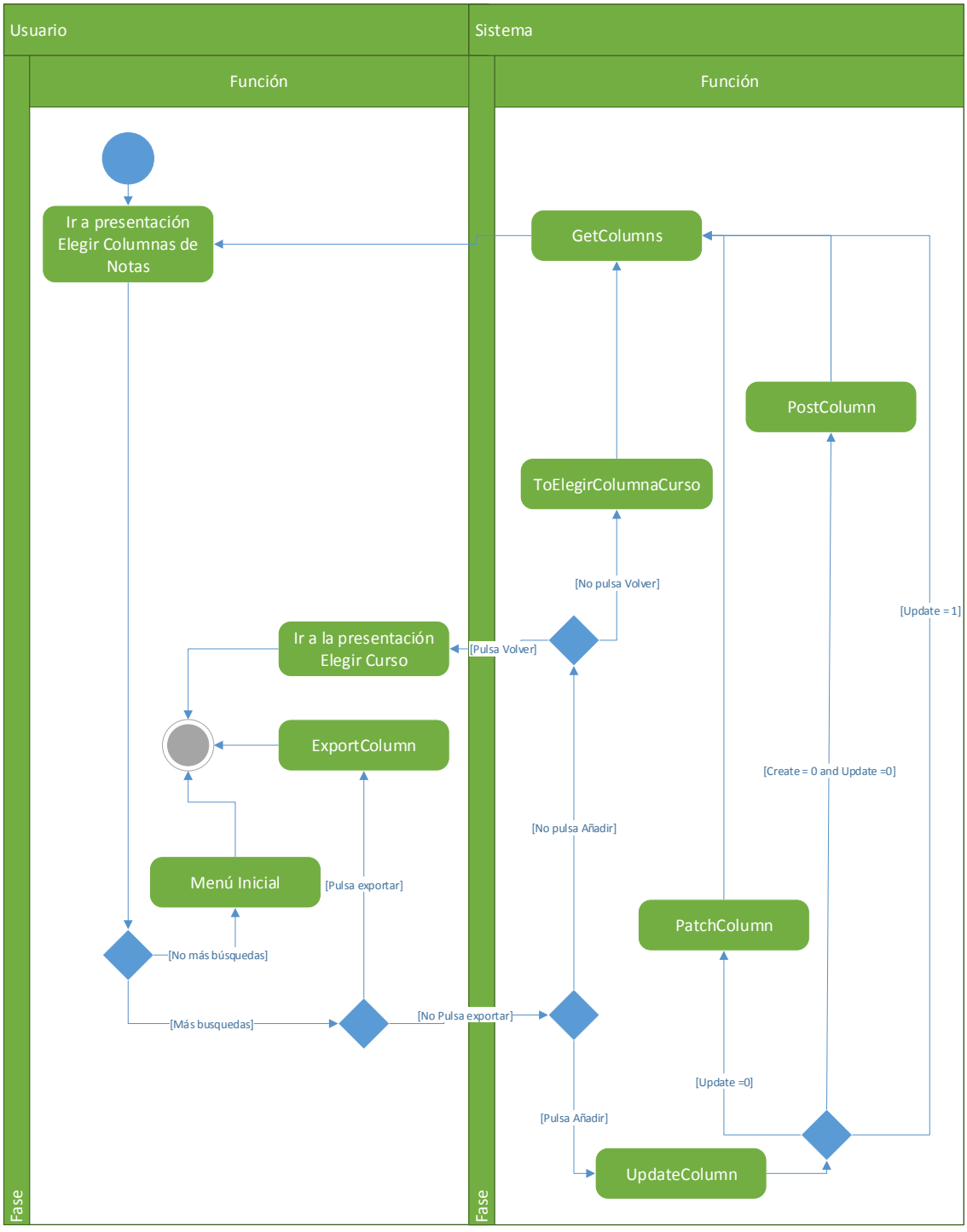


Diagrama de actividad 5. Elegir Columna Curso

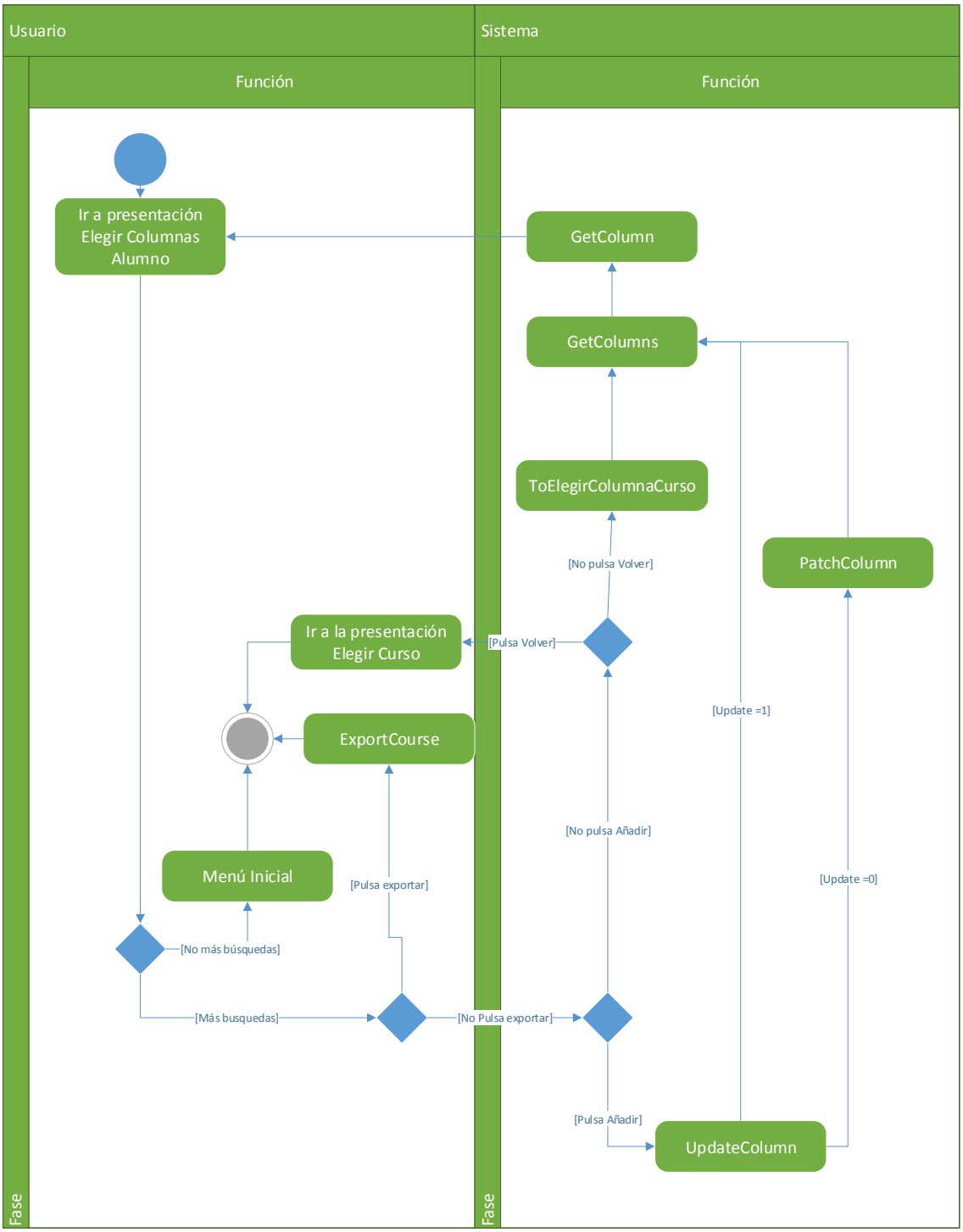


Diagrama de actividad 6. Elegir Columna Alumno

Diagrama de Clases

A continuación, se expone el diagrama de clases obtenido siguiendo las indicaciones de la API REST de Blackboard. El diagrama se ha orientado para ser usado en FileMaker, usando sus tipos de datos.

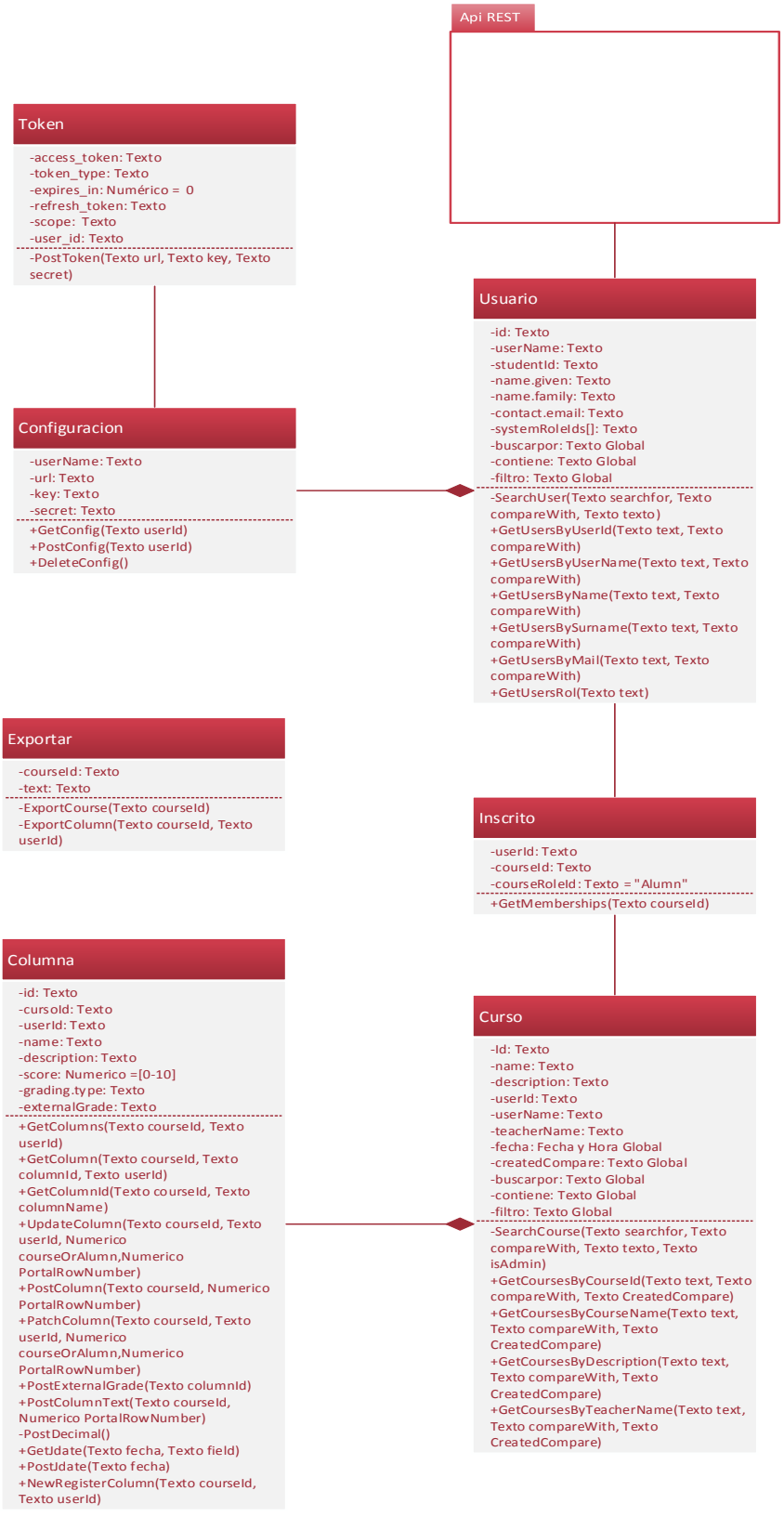


Diagrama de Clases 1.

Configuración de la Plataforma Blackboard Learn

Prerrequisitos

- Máquina virtual Linux con el paquete Blackboard Learn
    - bb-learn-Q2-2017-ga.zip que contiene:
        - bb-learn-9.1.Q2-2017.box
        - Vagrantfile
- El emulador de máquinas virtuales Virtual Box
- La aplicación Vagrant para lanzar la máquina virtual a partir de una plantilla

Máquina virtual Linux con el paquete Blackboard Learn

La copia de la máquina virtual estaba alojada en la página dedicada a desarrolladores y ha sido suministrada por el tutor de la empresa en un archivo zip llamado bb-learn-Q2-2017-ga.zip y que contenía la máquina virtual empaquetada por Vagrant bb-learn-9.1.Q2-2017.box y la plantilla usada por Vagrant para cargar la máquina virtual llamada Vagrantfile.

Instalación de Virtual Box

La instalación del programa ha sido correcta.

Instalación de Vagrant

Vagrant es una herramienta para la creación y configuración de entornos de desarrollo virtualizados. Originalmente se desarrolló para VirtualBox y sistemas de configuración tales como Chef, Salt y Puppet. Sin embargo, desde la versión 1.1 Vagrant es capaz de trabajar con múltiples proveedores, como VMware, Amazon EC2, LXC, DigitalOcean, etc. Aunque Vagrant se ha desarrollado en Ruby se puede usar en multitud de proyectos escritos en otros lenguajes, tales como PHP, Python, Java, C# y JavaScript.

La aplicación puede descargarse desde: Vagrant

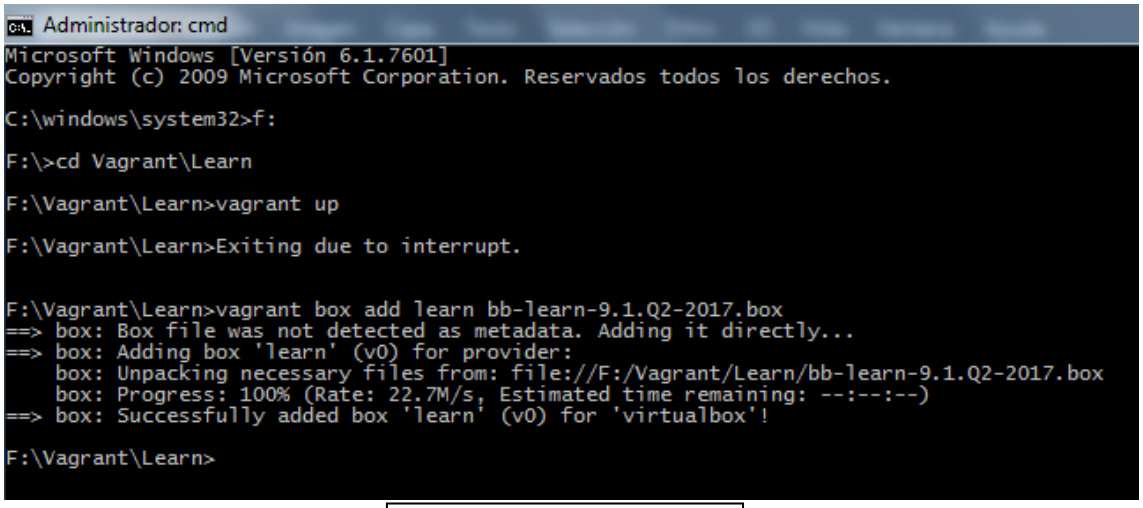
La instalación del programa ha sido correcta.

Carga de la máquina Virtual con Vagrant

Para cargar la máquina Virtual he creado en un pendrive la siguiente ruta unidad:\Vagrant\Learn y he descomprimido el contenido del archivo bb-learn-Q2-2017-ga.zip. Obteniendo los archivos bb-learn-9.1.Q2-2017.box y Vagrantfile con los que únicamente con ejecutar Vagrant up debería ser suficiente para cargar la máquina virtual con sus servicios.

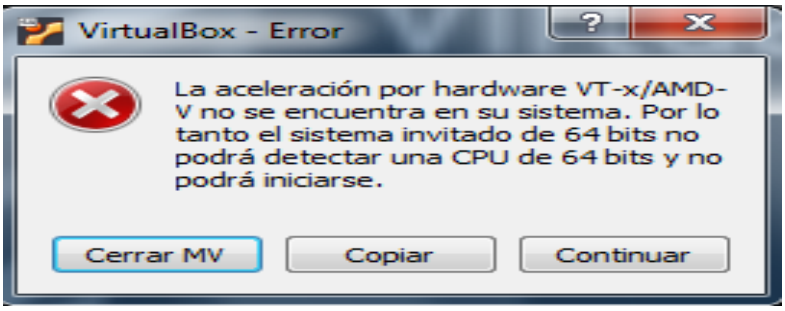
Sin embargo, al tratar de hacerlo en mi ordenador de sobremesa, me he encontrado con que Vagrant es incapaz de cargarla, ejecuté el comando Vagrant up y esperé cerca 2 horas porque el archivo era muy pesado y tal por eso tardaba tanto. Pero no ha hecho absolutamente nada, he matado el proceso sin que finalizara. Es como si no hubiera reconocido el contenido del archivo Vagrantfile.

Probé con el comando vagrant box add learn bb-learn-9.1.Q2-2017.box para que la máquina virtual se incluyera entre las máquinas que era capaz de cargar y aquí sí que el programa realizó tareas normalmente. Como se puede comprobar en la Captura 1.



Captura 1. vagrant box

Para intentar solucionar el problema y desde VirtualBox, he importado el archivo box.ovf que estaba en la ruta C:\Users\sergio\.vagrant.d\boxes\learn\0\virtualbox. La importación me indicaba que era una máquina Ubuntu de 64 bits y la importa de forma correcta, con la configuración que venía por defecto. Le doy a arrancar y me lanza el mensaje de error que aparece en la Captura 2.



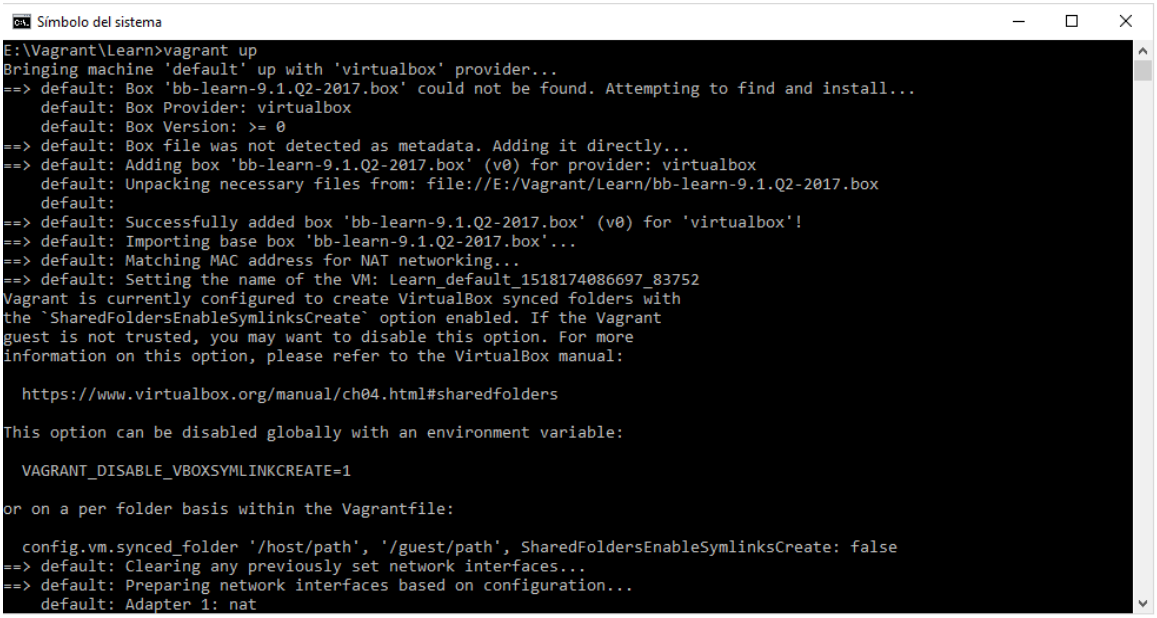
Captura 2. Error de VirtualBox

Según parece, el problema es mi procesador, un Intel® Core™2 Duo E4600, que, a pesar de admitir operaciones de 64 bits, no soporta virtualización en 64 bits. He entrado en la página de Intel para comprobarlo (E4600), en el apartado Tecnología de virtualización Intel® (VT-x), pone que no es soportado.

He repetido los mismos pasos en mi ordenador portátil y me he encontrado con el mismo problema, ya que es un Intel® Pentium® T4200, que, a pesar de admitir operaciones de 64 bits, tampoco soporta virtualización en 64 bits. He entrado en la página de Intel para comprobarlo (T4200), en el apartado Tecnología de virtualización Intel® (VT-x), pone que no es soportado.

He repetido los pasos en un ordenador del aula L130 de la Universidad de la Rioja y por fin la máquina virtual se ha cargado correctamente, ya que es un Intel® Core™ i5-6600, procesador que admite operaciones de 64 bits y adicionalmente, soporta virtualización en 64 bits. He entrado en la página de Intel para comprobarlo (i5-6600), en el apartado Tecnología de virtualización Intel® (VT-x), pone que es soportada.

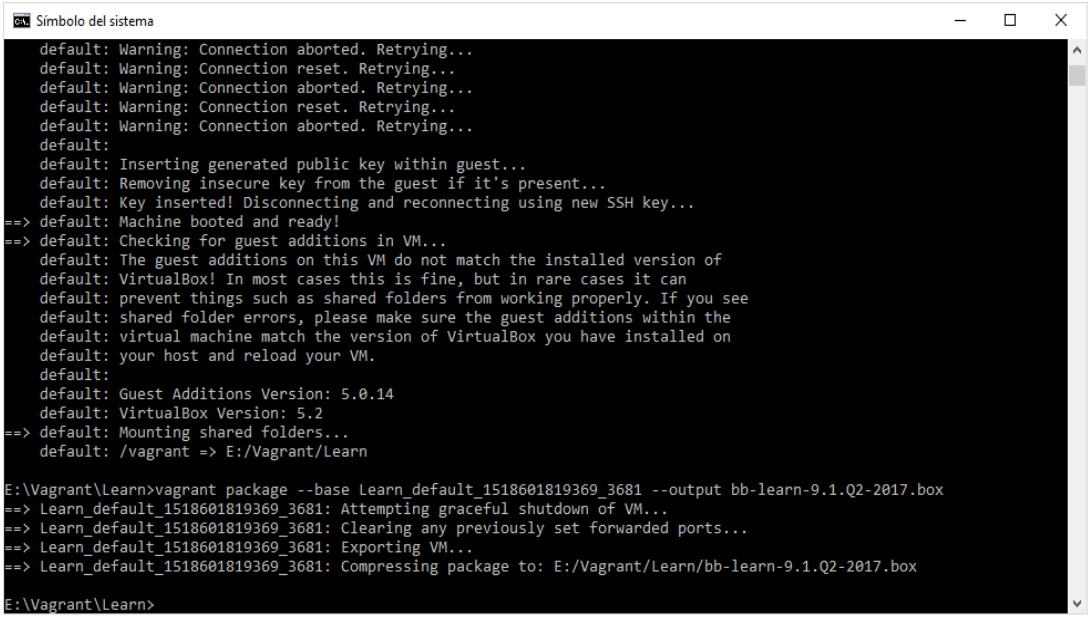
En la Captura 3 se puede comprobar cómo la máquina virtual se carga automáticamente al utilizar el comando vagrant up, sin que se necesite realizar ninguna tarea adicional.



Captura 3. vagrant up

Cuando vagrant carga una máquina, crea una nueva carpeta con un número único, de esta manera pueden guardarse varias versiones de la misma máquina, distinguiéndolas por carpetas numeradas; esta característica de vagrant no me ha resultado útil porque al apagar el ordenador del aula, perdía todas las versiones. Para poder seguir trabajando otro día con los cambios, es preciso que realice un backup de la misma.

He usado el comando vagrant package –base <nombre de la máquina virtual> --output <nombre del box destino>. El nombre de la máquina virtual puede obtenerse dentro de directorio del sistema C:\Users\<usuario>\VirtualBox VMs, en mi caso se llamaba Learn_default_1518681819369_3681. El nombre del box de destino puede ser el mismo que el original, lo sobrescribe si ya existe y si no lo crea. En la Captura 4 puede verse el resultado del comando con la máquina que quería guardar en el pendrive.



Captura 4. vagrant package

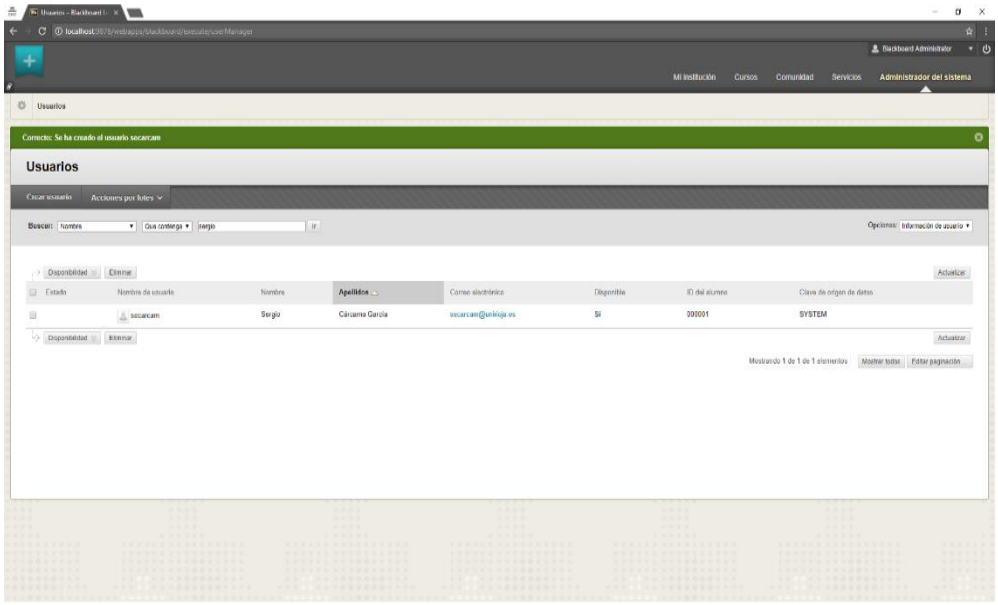
Configuración de Blackboard Learn

Blackboard Learn es un entorno de aprendizaje virtual y un sistema de gestión de cursos desarrollado por Blackboard Inc. Es un software de servidor basado en la web que presenta gestión de cursos, arquitectura abierta personalizable y diseño escalable que permite la integración con la información del alumno, sistemas y protocolos de autenticación. Puede instalarse en servidores locales o ser hospedado por Blackboard ASP Solutions. Su propósito principal es agregar elementos en línea a cursos impartidos tradicionalmente en persona y desarrollar cursos completamente en línea con poca o ninguna asistencia en persona y cara a cara.

Para poder realizar más adelante las pruebas y test, se van a crear al menos siete cursos, un profesor por cada curso y diez alumnos, de los cuales cinco van a ser hombres y cinco mujeres. Esos alumnos serán reutilizados para poblar los cursos; en uno se usarán todos, en otro sólo habrá mujeres, en otro sólo habrá hombres, en otro con solo mujeres se incluirá una mujer que además es profesor de otro curso, en otro con solo hombres se incluirá un hombre que además es profesor de otro curso, en otro se incluirá un profesor que además es administrador con el resto de alumnos y en el último se incluirá un profesor que es administrador y un alumno que es profesor de otro curso con el resto de alumnos.

Antes de crear el curso y usando el superusuario Administrator, voy a crear un nuevo usuario que tendrá el rol alumno y me servirá como referencia para saber cómo se crean los usuarios de forma genérica. Para ello, entro al Administrador del Sistema y en el submenú Usuarios, entro en usuarios, pulso el botón Crear usuario, nuevo y relleno todos los datos obligatorios, además de correo electrónico (con el formato, nombre@unirioja.es), un ID único de usuario (seis dígitos, empezando en 1 y rellenando con 0 a su izquierda), en este caso 000001, por ser el primero el sexo, una fecha de nacimiento, un código postal, un estado/provincia, el país, dejando el rol Alumno por defecto. Para confirmar, pulso el botón enviar.

Para comprobar que se ha creado correctamente voy a Administración del Sistema, dentro del submenú usuario, usuario y dentro del formulario buscar, pongo buscar por nombre, dejo la opción que contenga, y en el cuadro de texto escribo el nombre que le asigné, pulsando después el botón Ir. En la Captura 5, se puede ver que mi usuario de prueba se ha generado correctamente.



Captura 5. comprobación de mi usuario

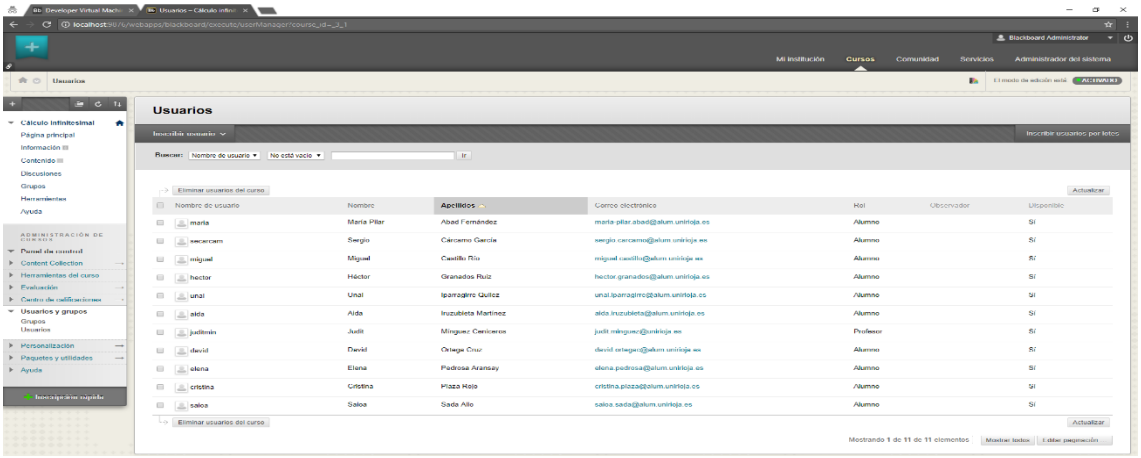
Para crear un usuario profesor se realizan los mismos pasos, la única diferencia notable es la elección del rol, he de asignarle el de Profesor.

Realizo la misma operación hasta completar los diez alumnos, añadiendo cinco hombres más y cinco mujeres.

Ahora hay que crear el curso, estando logeado como Administrator, lo creo. Para ello, entro al Administrador del Sistema y en el submenú Cursos, entro en cursos, pulso el botón Crear curso y relleno todos los datos obligatorios, usando el ID curso 814. Para confirmar, pulso el botón enviar.

Ahora hay que asignarle un profesor y alumnos, para ello entro en Administrador del Sistema y en el submenú Cursos y en el formulario buscar, dejo nombre del curso. Aparece el curso que acabo de crear, lo selecciono pulsando al lado de su ID, se despliega un menú, selecciono inscripciones, pulso sobre inscribir usuarios. Me aparece un formulario que permite inscribir por rol, lo cambio a profesor, pulso sobre examinar y selecciono el profesor, uso el botón Enviar para confirmar. Repito la operación, pero esta vez pongo el rol de alumno y al pulsar en examinar, en vez de seleccionar sólo un alumno, selecciono todos los que me interesan, pulso sobre el botón de Enviar para confirmar.

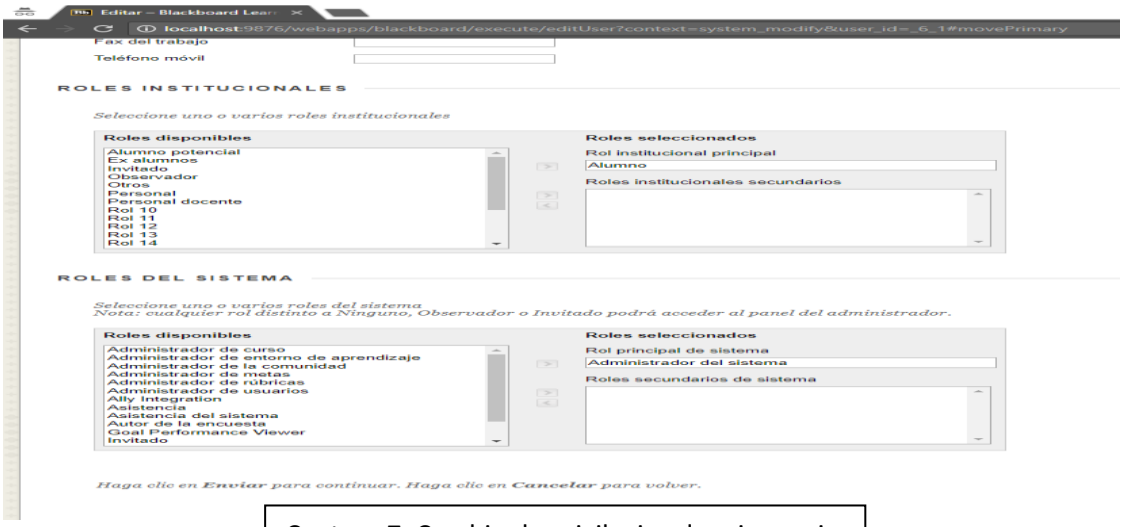
Ahora ya tengo un curso con un profesor asignado y diez alumnos. Para comprobarlo, entro en Administrador del Sistema y en el submenú Cursos y en el formulario buscar, dejo nombre del curso y selecciono la opción no está vació. Aparece el curso que acabo de crear, lo selecciono pulsando al lado de su ID, se despliega un menú, selecciono abrir y en menú de la izquierda pulso sobre usuarios y grupos para desplegar el menú, selecciono usuarios. Me aparece la lista de la Captura 6, en la cual se puede ver que el curso se ha generado correctamente y que tiene asignado su profesor y sus alumnos.



Captura 6. comprobación de cursos con profesor y alumnos

Repito la operación de creación de cursos, hasta completar los siete casos distintos que había preparado para las pruebas. Para poder realizar pruebas y test necesito tener un alumno o un profesor con privilegios de Administrador de sistema. Elijo un alumno y asciendo sus privilegios asignándole el rol del sistema de Administrador del sistema. Este paso es necesario porque necesitamos un usuario con privilegios de superusuario dentro del API Rest y sería inseguro asignárselo al usuario por defecto Administrator.

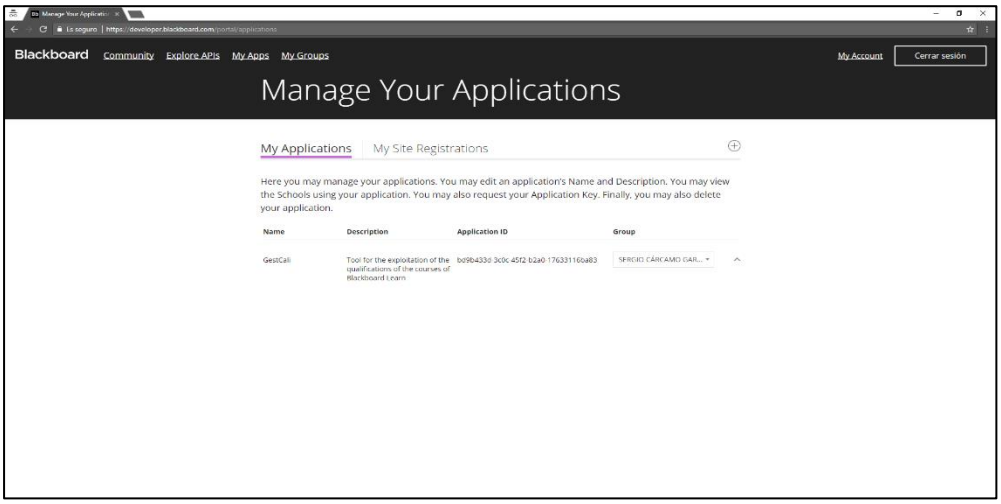
Entro en Administrador del Sistema y en el submenú Usuarios y en el formulario buscar, dejo nombre del usuario. Aparecen todos los usuarios que se han creado, elijo el usuario, selecciono al lado del nombre del usuario y elijo la opción del menú desplegable editar. Dentro de los datos del usuario voy a Roles del sistema, selecciono Administrador del sistema y pulso la pestaña de Rol principal del sistema. Como puede verse en la Captura 7. Para que los cambios sean efectivos pulso el botón enviar.



Captura 7. Cambio de privilegios de mi usuario

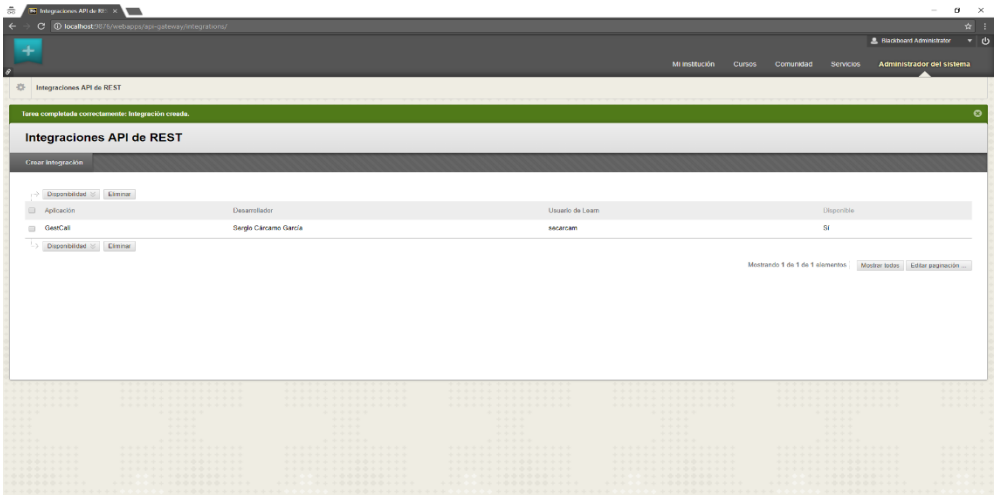
Habilitar la instancia REST en Blackboard Learn

Para habilitar la instancia REST en Blackboard Learn es necesario tener registrado un usuario desarrollador en la página de Blackboard. Una vez que dispongamos de un usuario registrado, debemos añadir la aplicación (en la sección My Apps, pulsamos sobre el botón + y rellenamos los formularios, dándole un nombre y una descripción), tras lo cual, se nos asignará un ID de aplicación único (Application ID), una llave de aplicación (Application Key) y una cadena secreta (Secret), es aconsejable guardar en sitio seguro esos valores. En la Captura 8 puede verse el resultado de añadir la aplicación.



Captura 8. Aplicación añadida

Vamos al menú Administrador del sistema, dentro del submenú Building Blocks, entramos en Integraciones Api REST, pulsamos el botón crear integración. Nos aparece un formulario en el debemos añadir de forma obligatoria un ID de aplicación y un usuario. Introducimos el ID de nuestra aplicación (aunque todavía no está implementada) y pulsando el botón Examinar buscamos por el nombre del usuario que previamente habíamos asignado como Administrador de sistema para pruebas, ya que para la aplicación ese usuario es el desarrollador responsable de la misma. Para confirmar pulsamos en el botón Enviar. En la Captura 9 puede verse la aplicación ya integrada.



Captura 9. Aplicación integrada

Estudio del uso de Java para consumir el servicio REST

Para desarrollar la aplicación se eligió estudiar la posibilidad de utilizar el lenguaje de programación Java porque en la documentación de Blackboard Learn dice que es compatible con sus servicios REST (también permite el uso de C#, Phyton, Ruby, PHP, CURL y Goland), hay diversos ejemplos útiles, es un lenguaje que conozco y no es necesario un aprendizaje previo.

Requisitos para desarrollar el servicio REST

- Máquina Virtual de Blackboard Learn con la instancia REST habilitada
- Máquina virtual de Java
- Servidor Tomcat
- Entorno de desarrollo para escribir aplicaciones en Java:
  - Eclipse OXYGEN EE Developers
- API Java JAX-RS:
  - Implementación Jersey
- Drivers JDBC para FileMaker

Las llamadas a REST serían gestionadas desde Java apoyándose en la implementación Jersey para posteriormente ser procesados los Json recibidos, usando JDBC como pasarela y mediante instrucciones SQL, los resultados serían mostrados al usuario desde FileMaker. Cualquier modificación del usuario volvería como instrucciones SQL usando JDBC como pasarela, de vuelta a la aplicación escrita en Java y apoyándose en la implementación Jersey, actualizaría la información que enviará en forma de Json, de regreso al servicio REST de Blackboard.

Después de comprobar que era posible realizar una interacción con los servicios REST desde Java, al compartir impresiones con el tutor de la empresa, se decidió que, si se iba a utilizar como consumidor final FileMaker, la programación en Java iba a ser redundante y, por tanto, se abandonó esta posibilidad, provocando una alteración en la planificación, el diseño y el tiempo disponible.

Diseño de los formularios en FileMaker

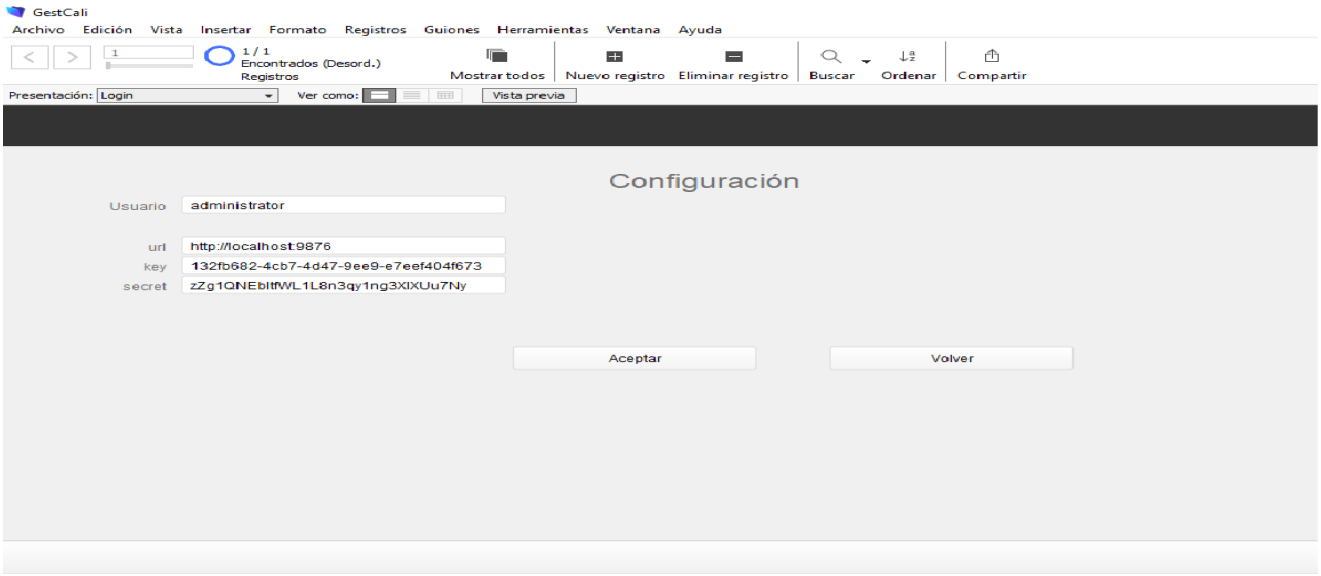
A continuación, se van a enumerar los distintos formularios en FileMaker que componen la aplicación.

En el Formulario 1 puede verse un menú emergente que permite filtrar la búsqueda del usuario mediante los desplegables Buscar (nos da las opciones de: ID de alumno, Nombre de usuario, Nombre, Apellidos y Correo electrónico) y otro para las opciones Que contiene, Igual a y Comienza por. Los resultados se obtienen cuando el usuario escribe en el cuadro de texto y pulsa el botón Buscar y se muestran por filas dentro del portal, dando información sobre el Nombre de Usuario, Nombre, Apellidos, Email e ID del alumno. También da la posibilidad de elegir ese usuario para pasar al formulario de elección de cursos al pulsar sobre el botón Cursos.

En caso de que el usuario no quiera realizar más operaciones, tiene disponible en la parte inferior derecha, el botón Menú inicial, que regresa al Menú inicial, borrando los resultados obtenidos.

Formulario 1. Elección de usuario

En el Formulario 2 pueden verse los datos necesarios para realizar una conexión con el API Rest. Un botón Aceptar que nos devolverá al Menú de Inicial guardando los cambios y un botón volver que nos mandará al Menú inicial restaurando los valores previos. Este formulario sólo podrá ser accedido desde el Menú inicial y podrá manipularlo cualquier usuario. Se trata de datos críticos para la aplicación, por lo que el usuario es el responsable de introducir los datos correctos que permitan la consulta contra el API Rest.

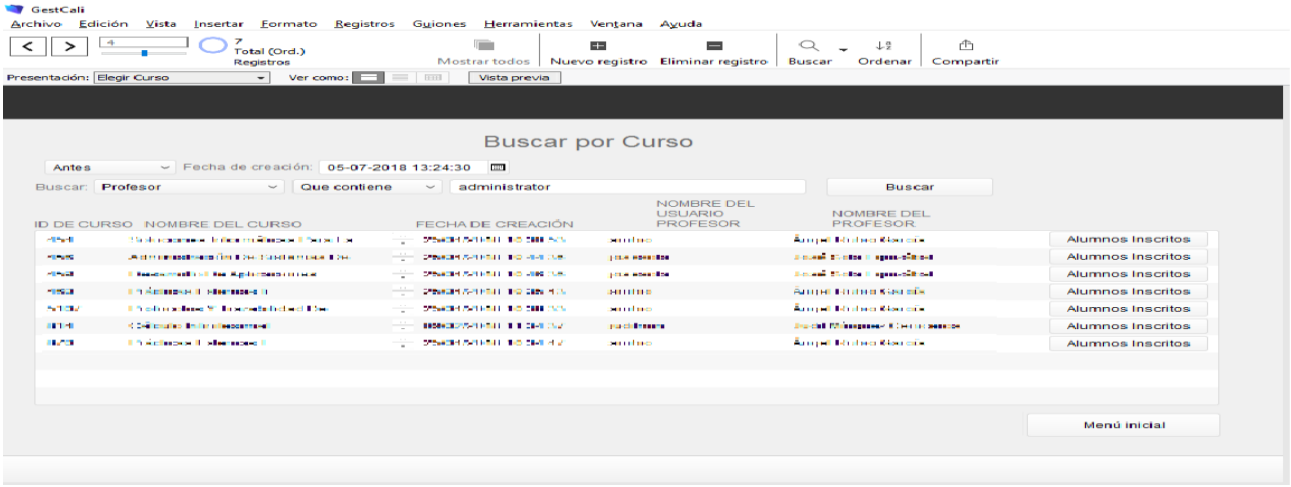


Formulario 2. Configuración

En el Formulario 3 puede verse un menú emergente que permite filtrar la búsqueda por fechas (nos da las opciones Después y Antes) acompañado por un textBox que por defecto marca la fecha actual y un calendario desplegable, para que el usuario pueda elegir una nueva.

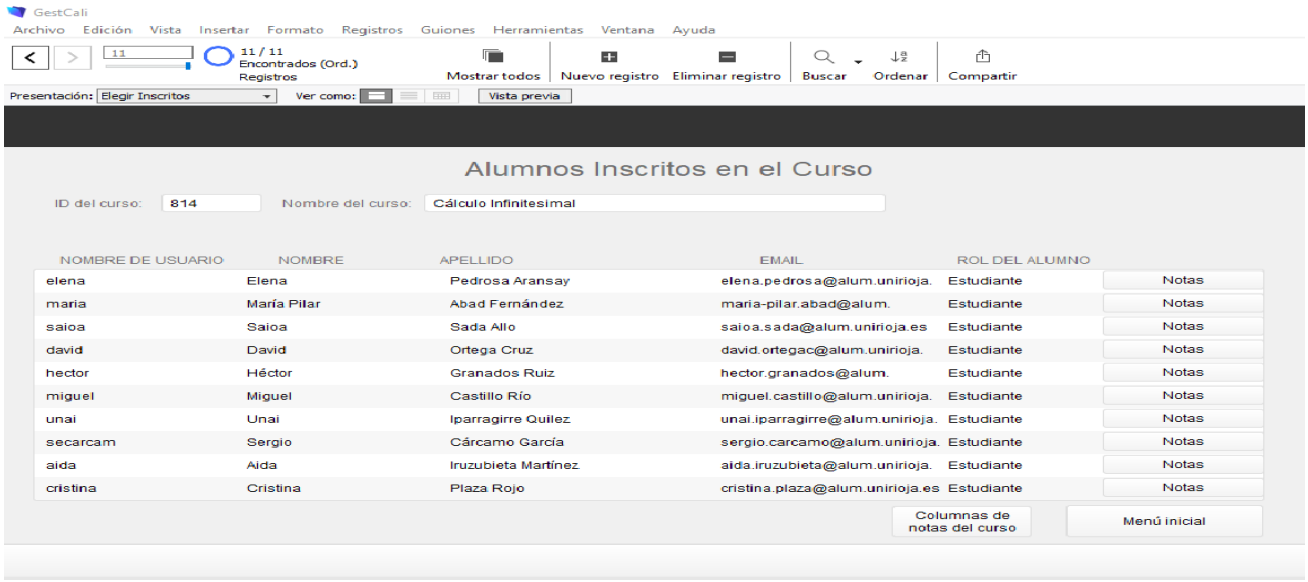
Un menú emergente para Buscar (nos da las opciones de: ID de alumno, Nombre de usuario, Nombre, Apellidos y Correo electrónico) y otro para las opciones Que contiene, Igual a y Comienza por. Los resultados se obtienen cuando el usuario escribe en el cuadro de texto y pulsa el botón Buscar y se muestran por filas dentro del portal, dando información sobre el ID del curso, Nombre del curso, Fecha de Creación, Nombre de Usuario del profesor y Nombre del Profesor También da la posibilidad de elegir ese curso para pasar al formulario de elección de alumnos inscritos al pulsar sobre el botón Alumnos inscritos.

En caso de que el usuario no quiera realizar más operaciones, tiene disponible en la parte inferior derecha, el botón Menú Inicial, que regresa al Menú inicial, borrando los resultados obtenidos.



Formulario 3. Elección de Curso

En el Formulario 4 puede verse una cabecera que muestra el Id del Curso y el Nombre del curso. Y debajo un portal con los datos para cada alumno de su Nombre de usuario, Nombre, Apellido, Email y Rol del Alumno (sólo puede ser Estudiante). También da la posibilidad de elegir ese alumno para pasar al formulario de elección de columnas al pulsar sobre el botón Notas.



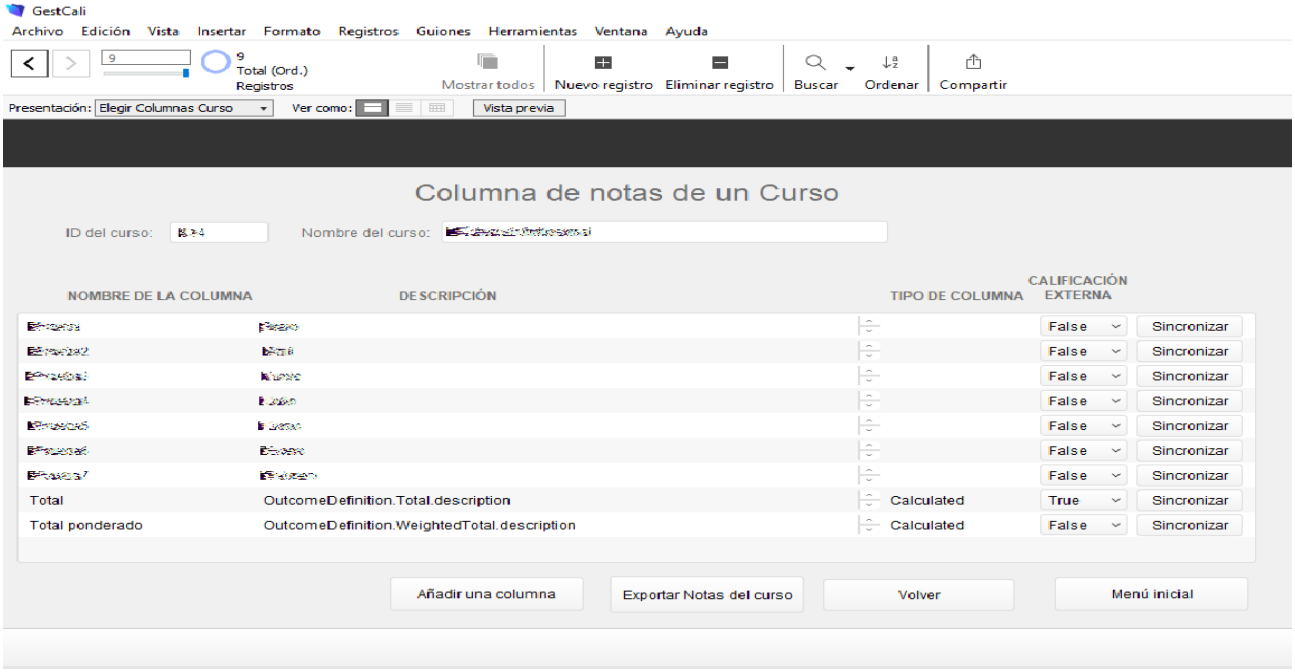
Formulario 4. Elección de alumno inscrito en un curso

En caso de que el usuario quiera consultar o modificar las columnas de notas del curso, tiene disponible en la parte inferior derecha y a la izquierda, el botón Columnas de notas del curso.

En caso de que el usuario no quiera realizar más operaciones, tiene disponible, en la parte inferior derecha y a la derecha, el botón Menú Inicial, que regresa al Menú inicial, borrando los resultados obtenidos.

En el Formulario 6 puede verse una cabecera que muestra el Id del Curso y el Nombre del curso. Y debajo un portal con los datos para cada columna de notas, con su Nombre de columna, Descripción, Tipo de columna (calculada si aparece como Calculated, en otro caso, el campo aparece vacío), un menú desplagable llamado Calificación Externa (puede valer True o False). También da la posibilidad de actualizar esa columna al pulsar el botón Sincronizar.

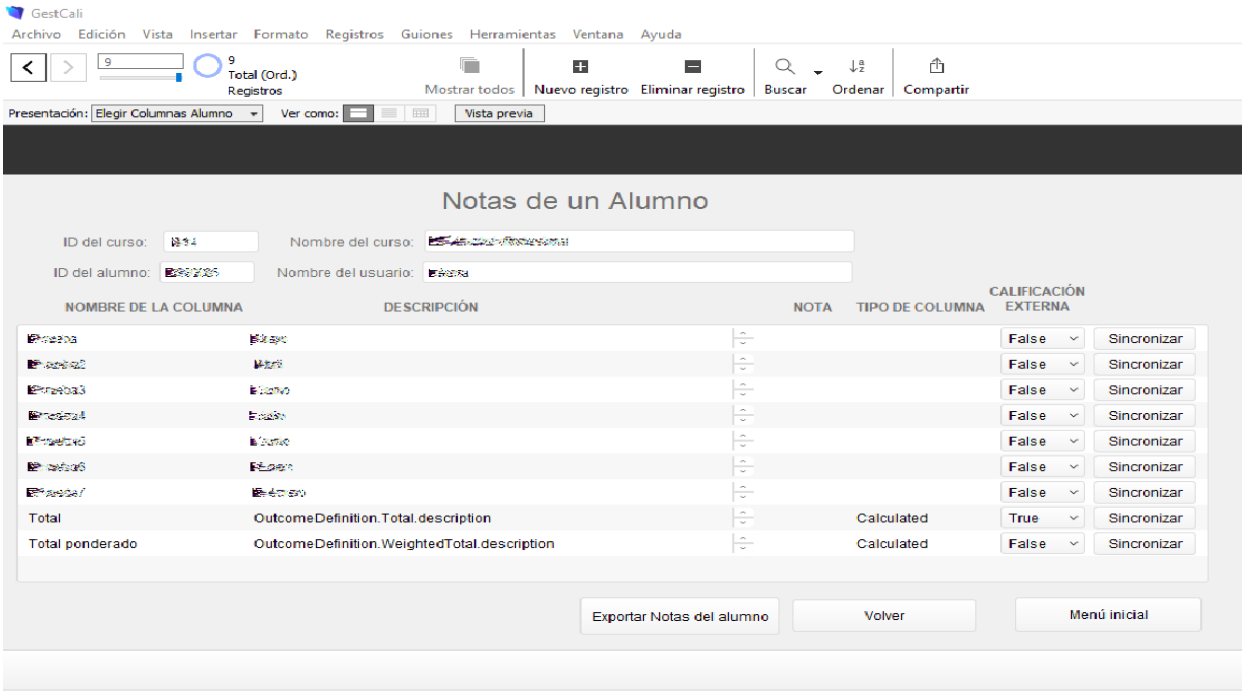
En la parte inferior hay disponibles cuatro botones, el primero, llamado Añadir columna, permite crear y modificar una columna nueva. El segundo, llamado Exportar Notas del curso, permite la exportación de todas las notas del curso agrupadas por alumno. El tercero, llamado Volver, permite regresar al formulario de Elección de alumno. El cuarto y último permite al usuario regresar al Menú inicial, borrando los resultados obtenidos.



Formulario 5. Elección de las columnas de notas

En el Formulario 6 puede verse una cabecera que muestra el Id del Curso, el Nombre del curso, el Id del alumno y el Nombre del Usuario. Y debajo un portal con los datos para cada alumno de su Nombre de columna de notas, Descripción, Nota, Tipo de columna (calculada si aparece como Calculated, en otro caso, el campo aparece vacío), un menú desplagable llamado Calificación Externa (puede valer True o False). También da la posibilidad de actualizar esa columna al pulsar el botón Sincronizar.

En la parte inferior hay disponibles tres botones, el primero, llamado Exportar Notas del alumno, permite la exportación de las notas del alumno visible en el portal. El segundo, llamado Volver, permite regresar al formulario de Elección de alumno. El tercero y último permite al usuario regresar al Menú inicial, borrando los resultados obtenidos.



Formulario 6. Elección de las columnas de notas de un alumno

Implementación de la funcionalidad en FileMaker

FileMaker utiliza un sistema de guiones para posicionarse sobre los distintos registros y distingue cada tabla como una presentación diferenciada, para reutilizar tablas utiliza un sistema de copia de tablas que llama autorelación.

En todo momento se ha optado por usar el modo presentación, también permite usar los modos búsqueda y vista previa, pero se ha decidido no utilizarlos, para no confundir al usuario y tener mayor control de la aplicación.

Para recoger la información desde el API REST, se ha incluido un guion de configuración que contiene los datos con el nombre usuario que va a manejar la aplicación, el cuál ha de existir en el sistema; la dirección URL del servidor de Blackboard Learn, se ha controlado si es http y su puerto 9876 o https y su puerto 9877 (comprobando si el principio y el final de la cadena contiene esos valores, en caso contrario el textBox le aparece en rojo al usuario); el Key y el Secret que utiliza Blackboard para tener permisos de desarrollador.

Por defecto se toma el de Administrator, si no es capaz de realizar una conexión, muestra un mensaje de error al usuario y vuelve al menú inicial. Para poder modificar los datos de configuración por defecto, dentro del menú inicial, hay un botón llamado configuración que permite al usuario cambiarlo a su gusto.

Esto podría suponer, potencialmente, un problema de seguridad, ya que mediante el API Rest sólo se puede hacer un control muy limitado de los roles que hay en el sistema, lo ideal sería poder comparar los usuarios con su password para implementar un sistema de login. Blackboard gestiona las password, pero el Api, por seguridad, no las devuelve. Por eso, en la aplicación, lo que sí se ha implementado es la posibilidad de realizar cambios en la configuración, ya sea porque la url ha cambiado o es necesario cambiar el Key o el Secret utilizados en Blackboard, ya sea porque se ha migrado el servidor y la url ha cambiado o porque se han asignado nuevos permisos a otro desarrollador.

Se ha dotado a la aplicación de seguridad a nivel de fichero, se le ha asignado una contraseña (1234) al Usuario Admin, la cuál será requerida siempre, al arrancar la aplicación. Una vez dentro de la aplicación, el usuario es libre de modificarla, pero este punto es crítico, porque si se prescinde de ella o se modifica,

un usuario malicioso podría alterar en su beneficio, la configuración que nos da acceso al API de Blackboard, al tener acceso sin estar autorizado.

Una posible mejora de la aplicación sería un mayor control de usuarios a nivel de permisos del fichero, ya que FileMaker permite hacerlo mediante las funciones GET (AccountName), que devuelve el nombre del usuario autenticado y después aplicarle GET (AccountPrivilegeSetName), que devuelve una lista de los nombres de privilegios que tiene el usuario en el fichero actual. Y luego consultando una lista blanca de usuarios permitidos, comparando con los privilegios que sean precisos para dar o denegar el acceso al fichero.

Para cargar datos desde el API REST se han realizado guiones específicos nombrados con el prefijo Get para consultas, tanto para cargar desde el API, como para consultar en la base de datos local. Con el prefijo Post para crear un nuevo objeto en el API. Con el prefijo Path cuando se quiere modificar un objeto en el API.

Se utilizan unas variables globales que se cargan al conseguir conectar correctamente con el API y se cargan o descargan a partir de los guiones que he llamado GetConfig, PostConfig y DeleteConfig. Y se corresponden con los datos de la configuración: $$user, $$url, $$key y $$secret. Adicionalmente hay otras variables auxiliares, para tratar fechas llamada $$fecha, para controlar la creación de columnas de notas $$create y para controlar la modificación de columnas de notas $$update.

En la carga de los datos se utiliza, dentro de los guiones la función de FileMaker, Insertar desde URL, la cual recibe los parámetros destino, que puede ser una variable o un campo de una tabla, como se reciben datos de un Json, los guardo en variables locales; ruta URL, aquí construyo una cadena a partir de la variable global $$url, $$token y la ruta que especifica el API para la operación especifica que preciso; especificar opciones de Curl, aquí añado cualquier cabecera adicional que se precise para obtener realizar la operación.

Para generar la variable $$token se hace una llamada al guion que he llamado POSTtoken que incluye una llamada a insertar URL y a partir de las variables globales $$url, $$key, $$secret se obtiene un token válido que será usado por el resto de guiones. Para evitar tener que estar comprobando cada cierto tiempo si está caducado o no, antes de realizar cualquier petición al servicio Rest, se llama a este guion, así tenemos asegurado que el token es válido en todo momento, aunque puede ser ineficiente al hacer esas llamadas que, si seguía siendo válido, son redundantes.

Para introducir los datos leídos desde la variable local se ha utilizado la función de FileMaker JSONGetElement, que recibe como parámetros la variable con el Json y una cadena con la ruta dentro del Json con la variable que queremos guardar. Para guardar los datos en las tablas se ha usado la función Insertar resultado calculado, que admite como parámetros el campo de tabla dónde se va a guardar el resultado y el resultado del JSONGetElement que hemos realizado previamente.

Para moverse entre las distintas presentaciones y el resto de guiones de operaciones, he usado unos guiones con el prefijo To seguido de elegir y el nombre de una tabla, si es un menú; sólo el nombre de una tabla, si se va a hacer una consulta específica; o Login, si sólo va a volver al menú inicial.

Para controlar los nombres que aparecen en los menús emergentes se ha usado un campo de tipo global llamado filtro. Inicialmente utilicé un guion llamado NameList, que guardaba en la variable global $$name el último nombre que el usuario había seleccionado junto con un guion llamado RestoreName, que para que no se sobrescribiera la información en el registro de la última selección, restauraba el estado del registro antes de la selección. Se eliminó esta solución en favor del campo de tipo global porque al restaurar el registro antes de la selección, podía crear confusión al usuario de la aplicación, al no mostrar de forma clara cuál era su última selección.

Para salir de la aplicación hay un guion genérico, llamado Exit, que puede ser llamado desde un botón. Cierra totalmente FileMaker, se sugirió que este botón preguntara al usuario si quería sincronizar los datos al salir, pero no se ha implementado porque ya existen botones específicos de sincronizar en los

formularios Elegir Curso y Elegir Columna, con sus correspondientes avisos de validación por parte del usuario.

En ocasiones, dentro de un guion ha sido necesario llamar a otro, para lo que se ha utilizado la función de FileMaker Ejecutar guion, que admite tantos parámetros como queramos, separados por la concatenación de cadenas & "¶" &. Después, en el guion que recibe la llamada, se pueden recuperar esos parámetros utilizando las funciones GetValue (GET (ScriptParameter); índice) dónde índice es el número de parámetro introducido, que toma el valor del parámetro con el índice que le hemos indicado; y la función Establecer variable, con los parámetros nombre de la variable de destino y el valor que queremos guardar, en este último parámetro introduciremos la función GetValue. También podemos usar unas opciones similares con los botones, sólo que la selección del guion asociado y los parámetros se realizan desde la interfaz gráfica de FileMaker, sin necesidad de usar guiones intermedios para hacer la llamada a la función.

Para la implementación del sistema de tratamiento de errores, se muestran cuadros de diálogo que muestran un mensaje al usuario y detienen el guion en curso, para impedir la incoherencia de los datos. Se tienen en cuenta la mayoría de errores detectados al recibir un Json con el campo status con un código de error numérico, aunque en algunos casos se han ignorado si estaban dentro de un bucle, ya que podrían suponer una molestia al usuario porque los mensajes de error se mostrarían en cada iteración, procurando salir del mismo lo antes posible y mostrando sólo el primer mensaje de error.

Test y pruebas

Para realizar pruebas del servicio REST y poder implementar la aplicación en FileMaker, se va a utilizar el programa de línea de comandos CURL en su versión para Windows. Los resultados de las pruebas servirán para establecer qué comandos se utilizarán más adelante en los guiones usados en FileMaker ya que en la función Insertar desde URL, entre sus parámetros recibe un comando Curl y unas cabeceras específicas para su correcto funcionamiento.

El servicio REST

Utilizando un servicio REST la aplicación se divide en una serie de recursos los cuales son accesibles usando una URI (una URL lógica, ficticia no física, creada siguiendo algún patrón que la representa), no hay nada "físico" (no es un HTML, ni una imagen), hay código que sabe cómo tratar esas URIs.

Los recursos usan una interface HTTP estándar para la transferencia del estado entre el cliente y los recursos, utilizando únicamente las operaciones HTTP: **POST**, **GET**, **PUT**, **PATCH** y **DELETE**.

Las operaciones **POST** son utilizadas para crear el recurso en el sistema.

Las operaciones **GET** son utilizadas para leer el recurso del sistema.

Las operaciones **PUT** son utilizadas para actualizar el recurso completo en el sistema, en caso de no usar el recurso completo, los campos no usados son tratados como nulos.

Las operaciones **PATCH** son utilizadas para actualizar parte del recurso en el sistema, es decir, sólo uno o varios campos concretos, no todos.

Las operaciones **DELETE** son utilizadas para borrar el recurso del sistema.

Los recursos serán tratados como objetos de CURL, ya que FileMaker 16 Pro permite trabajar con dichos objetos con soporte para JSON, mediante las URIs especificadas por el API REST de Blackboard. Los mensajes de respuesta usarán los códigos de estado HTML, especificados por la RFC 2616 (actualmente obsoleto), y algunos fragmentos en los estándares RFC 2518 (también obsoleto), RFC 2817, RFC 2295 (experimental), RFC 2774 (también experimental) y RFC 4918; otros no están estandarizados, pero son comúnmente utilizados.

Los códigos de respuesta para los mensajes más utilizados por el API Rest (aunque los estándares permiten muchos más) son:

- **200 OK**
  Respuesta estándar para peticiones correctas. Este mensaje nos devolverá un Json correcto.

- **201 Created**
  La petición ha sido completada y ha resultado en la creación de un nuevo recurso. Este mensaje nos devolverá el Json correcto que acabamos de crear.

- **204 No Content**
  La petición se ha completado con éxito, pero su respuesta no tiene ningún contenido. Este mensaje normalmente será devuelto cuando hemos realizado con éxito una operación de borrado.

- **400 Bad Request**
  La solicitud contiene sintaxis errónea y no debería repetirse. Este mensaje aparece cuando hemos introducido una consulta errónea.

- **403 Forbidden**
  La solicitud era correcta, pero el servidor rehúsa responderla dado que el cliente no tiene los privilegios para hacerla. En contraste a una respuesta 401 No autorizado, la autenticación no haría la diferencia. Este mensaje aparece cuando normalmente cuando Blackboard detecta que no se tienen permisos suficientes para tratar el objeto como lectura / escritura.

- **404 Not Found**
  Recurso no encontrado. Se utiliza cuando el servidor web no encuentra la página o recurso solicitado. Este mensaje aparece cuando el objeto ha sido borrado o hemos introducido un valor que no existe.

Obtención del token

Blackboard Learn utiliza la especificación de seguridad OAuth 2.0, permite a un usuario de Blackboard compartir su información en el sitio A (proveedor de servicio) con el sitio B (llamado consumidor) sin compartir toda su identidad. OAuth es utilizado por desarrolladores, es un método de interactuar con datos protegidos y publicarlos. Este mecanismo es utilizado por compañías como Google, Facebook, Microsoft, Twitter y GitHub para permitir a los usuarios compartir información sobre sus cuentas con aplicaciones de terceros o sitios web.

Para la autenticación de la aplicación la especificación genera un token, que es una cadena de texto cifrada a partir de un KEY y un PASS únicos, que sólo son conocidos por el desarrollador y la aplicación que desarrolle. Dicho token permitirá a la aplicación comunicarse con Blackboard para generar consultas contra su base de datos.

Para las pruebas voy a usar http por el puerto 9876, pero por seguridad, en producción es aconsejable que usemos https, ya que la información va cifrada mediante un certificado digital de confianza de Blackboard y deberemos utilizar el puerto 9877.

Para la obtención del token usando Curl utilizo la KEY 132fb682-4cb7-4d47-9ee9-e7eef404f673 y el PASS zZg1QNEbItfWL1L8n3qy1ng3XlXUu7Ny. Usando el siguiente comando:

curl -k –X POST -u 132fb682-4cb7-4d47-9ee9-e7eef404f673:zZg1QNEbItfWL1L8n3qy1ng3XlXUu7Ny -d “grant_type=client_credentials” https://localhost:9877/learn/api/public/v1/oauth2/token

El token aparece en el campo ‘access_token’ del Json de respuesta:

{“access_token”:”MvC6nkQApDbjkCNrrMC6CVls57FRY8Zm”, “token_type”:”bearer”, “expires_in”:3599}

A partir de ahora y para realizar todas las pruebas voy a usar el token que he obtenido, el cual tiene una validez por defecto de 1 hora, pasado ese tiempo o bien debo generar uno nuevo o renovar el actual.

Hay que tener en cuenta que Blackboard tiene las siguientes limitaciones para los desarrolladores:

- Gestión de 150 Users (Usuarios)
- Gestión de 100 Courses (Cursos)
- Gestión de 1000 Enrollments (Inscripciones)
- 10,000 peticiones en un periodo de 24 horas

En la aplicación, esta consulta se ha realizado siempre antes de realizar otra que precise Curl, guardando el resultado del Json en los campos de una tabla llamada Token, la cual es consultada mediante el guion llamado POSTtoken.

Se ha optado por hacerlo así para no tener que crear un sistema de control del tiempo, ya que, como se ha dicho antes, la validez de un Token es de 1 hora, aproximadamente. De esta manera me aseguro que, en todo momento, el Token es válido, ya que en cada petición recibimos uno nuevo. A no ser que haya un fallo de conexión, ya sea por fallos en la red o que el key o el secret de la configuración sean incorrectos, en cuyo caso, ese error será gestionado desde el guion, mostrando un mensaje al usuario informándole.

Obtención de un listado de usuarios

En la aplicación se pueden realizar búsquedas de los usuarios mediante consultas predefinidas en los comandos del API, permite discriminar los campos según sean necesarios (se utiliza el parámetro fields para devolver sólo esos campos), por lo que voy a usar para obtener un listado de usuarios de forma genérica el siguiente comando de Curl:

```
curl –k –X GET -H “Authorization: Bearer MvC6nkQApDbjkCNrrMC6CVls57FRY8Zm” http://localhost:9876/learn/api/public/v1/users?fields=id,userName,studentId,name.given,name.family,contact.email,systemRoleIds
```

Desde FileMaker solo se usarán los campos que aparecen en el parámetro fields, separados por comas. El API REST no permite realizar subconsultas para obtener información cruzada, hay que realizar primero la consulta y obtener los resultados, procesarla, para después realizar una nueva consulta sólo con los datos que nos interesan cruzar. Esta acción se realizará tantas veces como subconsultas necesitemos realizar.

Ahora voy a hacer consultas de prueba específicas para las consultas de los usuarios cuyos resultados serán procesados y mostrados al usuario desde FileMaker.

En concreto, se ha usado el guion llamado SearchUser para discriminar entre los tipos de consultas relacionadas con la elección de usuarios, permitiendo un filtrado de los mismos más fino.

Obtención de un listado de usuarios a partir del id de estudiante

Esta consulta se realiza dentro del formulario Elegir Usuario, se ejecuta cuando marcamos en el desplegable Buscar: ID del alumno y pulsamos el botón buscar. Se precisa que el usuario haya introducido un valor correspondiente (en el textBox del formulario) a un id de estudiante (no hay que confundirlo con el id, ya que también existe, pero se guarda en campo distinto), en otro caso, devolverá un Json nulo ({"results":[]}), por lo que FileMaker mostrará un mensaje al usuario. También se mostrarán errores al usuario si se obtienen Json con el campo status con los valores 400 y 403.

La consulta necesaria es la misma que para el listado de usuarios. Esta consulta es ineficiente, ya que nos devuelve todos los usuarios, se ha optado a realizarlo así porque el API no la facilita de forma específica. Se ha discriminado desde FileMaker a partir de una variable llamada $text, que contiene el id del estudiante a buscar, el cual, el usuario ha introducido previamente (en el textBox del formulario) y que se comparará con el campo studentId obtenido en la consulta.

Adicionalmente, desde FileMaker, el usuario podrá filtrar este resultado para eliminar todos aquellos que no cumplan una de las opciones relacionadas con $text:

- Que contiene
- Igual a
- Comienza por

También estaba la opción de filtrar por No está vacío, pero se ha descartado porque siempre mostraba todos los usuarios y no parecía filtrar nada.

Obtención de un listado de usuarios a partir del nombre de usuario

Esta consulta se realiza dentro del formulario Elegir Usuario, se ejecuta cuando marcamos en el desplegable Buscar: Nombre de usuario y pulsamos el botón buscar. Se precisa que el usuario haya introducido un valor correspondiente (en el textBox del formulario) a un nombre de usuario (no hay que confundirlo con el nombre de la persona, ya que también existe, pero se guarda en campo distinto), en otro caso, devolverá un Json nulo ({"results":[]},), por lo que FileMaker mostrará un mensaje al usuario. También se mostrarán errores al usuario si se obtienen Json con el campo status con los valores 400 y 403.

La consulta necesaria es la misma que para el listado de usuarios, pero con el parámetro userName al que le pasamos el contenido de la variable $text, que contiene el nombre del usuario. Esta consulta es más eficiente que la que usamos para obtener el listado por id de usuario, ya que nos devuelve sólo los usuarios que en el campo userName contienen $text.

Al igual que para la obtención del listado de usuarios a partir del id de estudiante, el usuario puede filtrar por:

- Que contiene
- Igual a
- Comienza por

Obtención de un listado de usuarios a partir de su nombre

Esta consulta se realiza dentro del formulario Elegir Usuario, se ejecuta cuando marcamos en el desplegable Buscar: Nombre y pulsamos el botón buscar. Se precisa que el usuario haya introducido un valor correspondiente (en el textBox del formulario) a su nombre, (no hay que confundirlo con el nombre de usuario, ya que también existe, pero se guarda en campo distinto), en otro caso, devolverá un Json nulo ({"results":[]},), por lo que FileMaker mostrará un mensaje al usuario. También se mostrarán errores al usuario si se obtienen Json con el campo status con los valores 400 y 403.

La consulta necesaria es la misma que para el listado de usuarios. Esta consulta es ineficiente, ya que nos devuelve todos los usuarios, se ha optado a realizarlo así porque el API no la facilita de forma específica. Se ha discriminado desde FileMaker a partir de una variable llamada $text, que contiene el nombre a buscar, el cual, el usuario ha introducido previamente (en el textBox del formulario) y que se comparará con el campo name.given obtenido en la consulta.

Al igual que para la obtención del listado de usuarios a partir del id de estudiante, el usuario puede filtrar por:

- Que contiene
- Igual a
- Comienza por

Obtención de un listado de usuarios a partir del apellido

Esta consulta se realiza dentro del formulario Elegir Usuario, se ejecuta cuando marcamos en el desplegable Buscar: Apellido y pulsamos el botón buscar. Se precisa que el usuario haya introducido un valor correspondiente (en el textBox del formulario) a su apellido, en otro caso, devolverá un Json nulo ({"results":[]}), por lo que FileMaker mostrará un mensaje al usuario. También se mostrarán errores al usuario si se obtienen Json con el campo status con los valores 400 y 403.

La consulta necesaria es la misma que para el listado de usuarios, pero con el parámetro name.family al que le pasamos el contenido de la variable $text, que contiene el apellido. Esta consulta es más eficiente que la que usamos para obtener el listado por id de usuario, ya que nos devuelve sólo los usuarios que en el campo name.family contienen $text.

Al igual que para la obtención del listado de usuarios a partir del id de estudiante, el usuario puede filtrar por:

- Que contiene
- Igual a
- Comienza por

Obtención de un listado de usuarios a partir de su email

Esta consulta se realiza dentro del formulario Elegir Usuario, se ejecuta cuando marcamos en el desplegable Buscar: Correo Electrónico y pulsamos el botón buscar. Se precisa que el usuario haya introducido un valor correspondiente (en el textBox del formulario) a su email, en otro caso, devolverá un Json nulo ({"results":[]}), por lo que FileMaker mostrará un mensaje al usuario. También se mostrarán errores al usuario si se obtienen Json con el campo status con los valores 400 y 403.

La consulta necesaria es la misma que para el listado de usuarios. Esta consulta es ineficiente, ya que nos devuelve todos los usuarios, se ha optado a realizarlo así porque el API no la facilita de forma específica. Se ha discriminado desde FileMaker a partir de una variable llamada $text, que contiene el nombre a buscar, el cual, el usuario ha introducido previamente (en el textBox del formulario) y que se comparará con el campo contact.email obtenido en la consulta.

Al igual que para la obtención del listado de usuarios a partir del id de estudiante, el usuario puede filtrar por:

- Que contiene
- Igual a
- Comienza por

Obtención de un listado de cursos

Para obtener un listado de cursos de forma genérica utilizo el siguiente comando de Curl:

curl –k –X GET -H “Authorization: Bearer MvC6nkQApDbjkCNrrMC6CVls57FRY8Zm” http://localhost:9876/learn/api/public/v1/courses?fields=id,courseId,name,description,created

En concreto, se ha usado el guion llamado SearchCourse para discriminar entre los tipos de consultas relacionadas con la elección de cursos, permitiendo un filtrado de los mismos más fino.

Obtención de un listado de cursos a partir del ID del curso

Esta consulta se realiza dentro del formulario Elegir Curso, se ejecuta cuando marcamos en el desplegable Buscar: ID del curso y pulsamos el botón buscar. Se precisa que el usuario haya introducido un valor correspondiente (en el textBox del formulario) a un id de curso (no hay que confundirlo con el id, ya que también existe, pero se guarda en campo distinto), en otro caso, devolverá un Json nulo ({"results":[]}), por lo que FileMaker mostrará un mensaje al usuario. También se mostrarán errores al usuario si se obtiene un Json con el campo status con el valor 400.

La consulta necesaria es la misma que para el listado de cursos, pero con el parámetro courseId al que le pasamos el contenido de la variable $text, que contiene el id del curso. Esta consulta es más eficiente que la que usamos para obtener el listado de cursos, ya que nos devuelve sólo los usuarios que en el campo courseId contienen $text.

Se precisa una subconsulta para poder mostrar al usuario, los datos relacionados con el profesor del curso, nombre de usuario, junto con su nombre y apellidos. Para la cuál utilizaremos el siguiente comando de Curl:

curl –k –X GET -H "Authorization: Bearer MvC6nkQApDbjkCNrrMC6CVls57FRY8Zm" http://localhost:9876/learn/api/public/v1//courses/<courseId>/users?fields=userId,courseId,courseRoleId

Donde <courseId> corresponde al contenido de la variable local $courseId. Esta subconsulta será procesada desde FileMaker a partir del campo courseRolId. Si contiene el valor Instructor, se realizará una nueva subconsulta del usuario utilizando el siguiente comando de Curl:

curl –k –X GET -H "Authorization: Bearer MvC6nkQApDbjkCNrrMC6CVls57FRY8Zm" http://localhost:9876/ learn/api/public/v1/users/<userId>?fields=id,userName,name.given,name.family

Donde <userId> será el contenido del campo userId obtenido en la subconsulta de usuario.

Guardaremos el contenido de los campos userName, para mostrar el nombre de usuario y concatenaremos name.given, name.family, para mostrar el nombre y apellidos en un único campo. También se mostrarán errores al usuario si se obtienen Json con el campo status con los valores 400 y 403.

Adicionalmente, desde FileMaker, el usuario podrá filtrar este resultado para eliminar todos aquellos que no cumplan una de las opciones relacionadas con $text:

- Que contiene
- Igual a
- Comienza por

También estaba la opción de filtrar por No está vacío, pero se ha descartado porque siempre mostraba todos los usuarios y no parecía filtrar nada.

A parte, se puede filtrar por la fecha de creación del curso, el usuario podrá filtrar este resultado para eliminar todos aquellos que no cumplan una de las opciones relacionadas con $fecha:

- Después
- Antes

Para gestionar las fechas desde FileMaker, se han utilizado dos guiones auxiliares llamados GetJdate (convierte la fecha obtenida desde una variable obtenida de un Json, guardándolo en el campo pasado como parámetro, que contiene un tipo Fecha y Hora de FileMaker) y PostJdate (convierte un campo de tipo Fecha y Hora, guardándolo en una variable global llamada $fecha con el formato que usa el API Rest para los Json de fecha).

Obtención de un listado de cursos a partir del nombre del curso

Esta consulta se realiza dentro del formulario Elegir Curso, se ejecuta cuando marcamos en el desplegable Buscar: Nombre del Curso y pulsamos el botón buscar. Se precisa que el usuario haya introducido un valor correspondiente (en el textBox del formulario) a un nombre de curso, en otro caso, devolverá un Json nulo ({"results":[]}), por lo que FileMaker mostrará un mensaje al usuario. También se mostrarán errores al usuario si se obtiene un Json con el campo status con el valor 400.

La consulta necesaria es la misma que para el listado de cursos, pero con el parámetro name al que le pasamos el contenido de la variable $text, que contiene el nombre del curso. Esta consulta es más eficiente que la que usamos para obtener el listado de cursos, ya que nos devuelve sólo los cursos que en el campo name contienen $text.

Para poder mostrar al usuario, los datos relacionados con el profesor del curso, nombre de usuario, junto con su nombre y apellidos, se utilizan las mismas subconsultas que se utilizaron para el listado de cursos a partir del ID de curso.

Al igual que para el listado de cursos a partir del ID del curso, permite filtrar por:

- Que contiene
- Igual a
- Comienza por

Y para las fechas:

- Después
- Antes

Obtención de un listado de cursos a partir de su descripción

Esta consulta se realiza dentro del formulario Elegir Curso, se ejecuta cuando marcamos en el desplegable Buscar: Descripción y pulsamos el botón buscar. Se precisa que el usuario haya introducido un valor correspondiente (en el textBox del formulario) a su descripción, en otro caso, devolverá un Json nulo ({"results":[]}), por lo que FileMaker mostrará un mensaje al usuario. También se mostrarán errores al usuario si se obtiene un Json con el campo status con el valor 400.

La consulta necesaria es la misma que para el listado de cursos, pero con el parámetro description al que le pasamos el contenido de la variable $text, que contiene la descripción del curso. Esta consulta es más eficiente que la que usamos para obtener el listado de cursos, ya que nos devuelve sólo los cursos que en el campo description contienen $text.

Para poder mostrar al usuario, los datos relacionados con el profesor del curso, nombre de usuario, junto con su nombre y apellidos, se utilizan las mismas subconsultas que se utilizaron para el listado de cursos a partir del ID de curso.

Al igual que para el listado de cursos a partir del ID del curso, permite filtrar por:

- Que contiene
- Igual a
- Comienza por

Y para las fechas:

- Después
- Antes

Obtención de un listado de cursos a partir de su profesor

Esta consulta se realiza dentro del formulario Elegir Curso, se ejecuta cuando marcamos en el desplegable Buscar: Profesor y pulsamos el botón buscar. Se precisa que el usuario haya introducido un valor correspondiente (en el textBox del formulario) a un profesor en el curso, pero en el caso de no ser encontrado, devolverá un Json nulo ({"results":[]}), por lo que FileMaker mostrará un mensaje al usuario. También se mostrarán errores al usuario si se obtiene un Json con el campo status con el valor 400.

En el caso de que se haya elegido un profesor previamente desde el formulario Elegir Usuario. Los filtros Buscar: Profesor y Que contiene, aparecerán por defecto y en el campo $text aparecerá el nombre de usuario correspondiente a ese profesor.

La consulta necesaria ya no es la misma que para el listado de cursos, esta vez se ha optado por filtrar primero por las fechas, usando el siguiente comando Curl:

curl –k –X GET -H “Authorization: Bearer MvC6nkQApDbjkCNrrMC6CVls57FRY8Zm” http://localhost:9876/learn/api/public/v1/courses?created=" & $$fecha & "&createdCompare=" & $createdCompare & "&fields=id,courseId,name,description,created

Donde $$fecha es una variable global que previamente ha sido tratada con el guion PostJdate para obtener una fecha válida en el API Rest. Y $createdCompare uno de los valores del filtro para fechas

Después (en el API es tratado como greaterOrEqual, que es el caso por defecto) o Antes (en el API es tratado como lessThan).

Ahora se filtran esos resultados desde FileMaker en función de si el usuario tenía el rol de Administrador o no, para lo que se usa el guion auxiliar GetUserRol. Este guion recibe como parámetro el contenido del campo $text y realiza una subconsulta del listado de un usuario con el parámetro userName, del cual obtenemos el campo systemRoleIds.

En el caso de que el primer valor (es un array que empieza en 0) del campo systemRoleIds sea SystemAdmin, entonces mostraremos al usuario todos los cursos obtenidos en la consulta por fechas. En caso contrario, realizaremos las mismas subconsultas que se utilizaron para el listado de cursos a partir del ID de curso.

Al igual que para el listado de cursos a partir del ID del curso, permite filtrar por:

- Que contiene
- Igual a
- Comienza por

Obtención de un listado de alumnos inscritos

Esta consulta se realiza dentro del formulario Elegir Curso, se ejecuta cuando pulsamos el botón Alumnos inscritos dentro de una fila del portal correspondiente a un curso, si devuelve un Json nulo ({"results":[]}), FileMaker mostrará un mensaje al usuario. También se mostrarán errores al usuario si se obtienen Json con el campo status con los valores 400, 403 y 404.

Para obtener un listado de los alumnos inscritos en el curso elegido utilizaremos el siguiente comando de Curl:

curl –k –X GET -H "Authorization: Bearer MvC6nkQApDbjkCNrrMC6CVls57FRY8Zm" http://localhost:9876/learn/api/public/v1/courses/<courseId>/users?fields=userId,courseId,courseRoleId

El parámetro <courseId> se corresponde con la variable local $courseId, cuyo valor se corresponde a un parámetro que habrá recibido el botón Alumnos inscritos. A partir de la consulta anterior, realizaremos la siguiente subconsulta, para cada ID de usuario obtenido, ya que representa la relación alumno que pertenece a un curso, para obtener los datos que mostraremos al usuario sobre los alumnos inscritos a un curso. Usando el siguiente comando Curl:

curl –k –X GET -H "Authorization: Bearer MvC6nkQApDbjkCNrrMC6CVls57FRY8Zm" http://localhost:9876/learn/api/public/v1/users/<userId>?fields=id,userName,studentId,name.given,name.family,contact.email,systemRoleId

Donde <userd> corresponde a la variable local $userId cuyo valor es obtenido de la lista obtenida con la consulta anterior y corresponde al campo userId.

Guardaremos el contenido de los campos userName para mostrar el nombre de usuario; name.given para mostrar su nombre; name.family, para mostrar sus apellidos; contact.email para mostrar su email y en función de que el campo courseRolId correspondiente al usuario sea Student (en el campo systemRoleId se mostrará al usuario como Estudiante) o sea Instructor (en el campo systemRoleId se mostrará al usuario como Profesor).

Obtención de un listado de las columnas de notas de un usuario

Esta consulta se realiza dentro del formulario Elegir Inscritos, se ejecuta cuando pulsamos el botón Notas dentro de una fila del portal correspondiente a un usuario, si devuelve un Json nulo ({"results":[]}), por lo que FileMaker mostrará un mensaje al usuario. También se mostrarán errores al usuario si se obtienen Json con el campo status con los valores 400, 403 y 404.

Para obtener un listado de las columnas de notas de un usuario elegido utilizaremos el siguiente comando de Curl:

curl –k –X GET -H "Authorization: Bearer MvC6nkQApDbjkCNrrMC6CVls57FRY8Zm" http://localhost:9876/learn/api/public/v1/courses/<courseId>/gradebook/columns/?fields=id

Se obtiene un listado de todos los Ids de columnas que pertenecen a un curso y ese valor será guardo en la variable local $columnId. Es necesaria una subconsulta que ha sido resuelta en un guion llamado GetColumn y que utiliza el siguiente comando Curl:

curl –k –X GET -H "Authorization: Bearer MvC6nkQApDbjkCNrrMC6CVls57FRY8Zm" http://localhost:9876/learn/api/public/v1/courses/<courseId>/gradebook/columns/<columnId>?fields=id,name,description,grading.type,externalGrade

Donde <courseId> sigue correspondiendo a la variable local $courseId y <columnId> a la variable local $columnId.

Guardaremos el contenido de los campos name para mostrar el nombre de la columna; description para mostrar la descripción de la columna; grading.type que mostrará al usuario si es un campo calculado (aparece como Calculated) o no (muestra el campo vacío); externalGrade que dice al usuario si es una nota externa o no, sólo puede existir uno activo y tiene los valores 1 o 0 (si es 1 se mostrará al usuario True y si es 0 se mostrará False), por defecto, la columna Total estará activa.

Es necesaria una subconsulta para obtener información de las notas y que utiliza el siguiente comando Curl:

curl –k –X GET -H "Authorization: Bearer MvC6nkQApDbjkCNrrMC6CVls57FRY8Zm" http://localhost:9876/learn/api/public/v1/courses/<courseId>/gradebook/columns/<columnId>/users/"&<userId>?fields=status,score

Donde <courseId> corresponde a la variable local $courseId, <columnId> corresponde a la variable local $columnId y <userId> corresponde a la variable local $userId. En otro caso, devolverá un Json nulo ({"results":[]}), por lo que FileMaker mostrará un mensaje al usuario. También se mostrará error al usuario si se obtiene un Json con el campo status con el valor 404.

Guardaremos el contenido del campo score para mostrar la nota de la columna al usuario.

Modificación de una columna de notas de un curso

Por defecto, en el sistema siempre tendremos las columnas llamadas Total ponderado y Total, las cuales pueden son del tipo Calculated, si el usuario trata de modificarlas, se le mostrará un mensaje advirtiéndole que no se puede. También se mostrarán errores al usuario si se obtienen Json con el campo status con los valores 400, 403 y 404.

Esta consulta de actualización se realiza cuando el usuario pulsa el botón añadir columna dentro del formulario Elegir columnas, después de crear una columna nueva o directamente cuando el usuario modifica una fila del portal que corresponde a una columna y pulsa sobre el botón sincronizar, se ha resuelto dentro del guion llamado UdateColumn y utilizando el guión auxiliar PatchColumn.

También se controla mediante el parámetro $courseOrAlumn si el formulario que hizo la llamada era Elegir Columnas Curso (tenía valor 1) en cuyo caso se permitirá modificar sólo el nombre de la columna (al crearla), su descripción y si era un externalGrade; o era Elegir Columnas Alumno (tenía valor 2) en cuyo caso se permitirá modificar sólo las notas y si era un externalGrade.

Se ha controlado mediante la variable global $$update, si el usuario ha pulsado el botón sincronizar después de crear una nueva columna (tiene el valor 0, hay una sincronización pendiente) o ha modificado una fila del portal (tiene el valor 1, se ha sincronizado correctamente). En caso de no haberlo hecho y pulsa el botón Volver o Menú inicial, pide confirmación al usuario para sincronizar en ese momento,

llamando al guion UpdateColumn. Se realiza la siguiente consulta de modificación mediante el comando de Curl:

curl –k -X PATCH -H “Authorization: Bearer MvC6nkQApDbjkCNrrMC6CVls57FRY8Zm” -H "Content-Type: application/json" –d @$jsonData http://localhost:9876/learn/api/public/v1/courses/<courseId>/gradebook/columns/<columnId>/users/<userId>

Donde <courseId> corresponde a la variable local $courseId, <columnId> corresponde a la variable $columnId y <userId> que corresponde a la variable local $userId. Se ha utilizado la cabecera de tipo Json (lo especificamos al decir que el Content-Type es application/json) que corresponde a una variable local de FileMaker llamada $jsonData. La cual se ha construido utilizando la función de FileMaker JSonSetElement, pasándole como parámetros la variable $jsonData y la tripleta [<nombre del campo>; <valor del campo>; <tipo del campo>].

El valor de la variable $columnId se obtiene a partir del guión getColumnId, que recibe como parámetros $courseId y $columnName mediante el siguiente comando Curl:

curl –k –X GET -H “Authorization: Bearer MvC6nkQApDbjkCNrrMC6CVls57FRY8Zm” http://localhost:9876/learn/api/public/v1/courses/<courseId>/gradebook/columns/?fields=id,name

En el caso de que se haya modificado el campo externalGrade en una fila del portal, se realiza una llamada al guión llamado PostExternalGrade y adicionalmente se realiza la siguiente subconsulta mediante el comando de Curl:

curl –k -X PATCH -H “Authorization: Bearer MvC6nkQApDbjkCNrrMC6CVls57FRY8Zm” -H "Content-Type: application/json" –d @$jsonData http://localhost:9876/learn/api/public/v1/courses/<courseId>/gradebook/columns/<columnId>

Donde <courseId> corresponde a la variable local $courseId y <columnId> corresponde a la variable $columnId. Se reutiliza la variable $jsonData, pero se reinicia y se guarda el valor externalGrade como True. Al usuario se le mostrarán los cambios poniendo la anterior columna que estaba activa a False y la elegida actualmente a True.

Si se ha actualizado correctamente, se mostrará un mensaje al usuario para informarle.

Creación de una columna de notas de un curso

Como ocurría con la modificación de una columna de notas de un curso, en el sistema siempre tendremos las columnas llamadas Total ponderado y Total y son tratadas de la misma manera.

Esta consulta de actualización se realiza cuando el usuario pulsa el botón añadir columna dentro del formulario Elegir Columnas Curso y se ha resuelto dentro del guion llamado UdateColumn y utilizando el guión auxiliar PostColumn. Se realiza siempre que las variables globales $$create y $$update tengan el valor 0. Su comportamiento es similar al de la modificación de la columna cuando no se ha sincronizado.

También se controla mediante el parámetro $courseOrAlumn si el formulario que hizo la llamada era Elegir Columnas Curso (tenía valor 1) en cuyo caso se permitirá modificar sólo el nombre de la columna, su descripción y si era un externalGrade. Para obtener el valor de la variable $columnId se realiza igual para la modificación de columnas.

La variable $jsonData se utiliza igual que como se ha explicado antes, pero en la creación se le pasan los siguientes valores por defecto: ["score.possible"; 10; JSONString] (la nota puede tendrá un valor entre 0 y 10), ["score.decimalPlaces"; 4; JSONString] (utiliza 4 posiciones decimales) y ["availability.available"; "Yes"; JSONString] (está lista para ser utilizada). De esta forma garantizamos que la nota sea válida al crearla y conforme a las reglas que se le exigen tanto en Blackboard como en FileMaker.

En el caso de que el valor del campo status sea 400, se mostrará un mensaje al usuario de que no ha sido posible la creación de la columna y por tanto tampoco se hará nada más.

Exportación de todas las notas de los alumnos de un curso

Esta consulta se realiza dentro del formulario Elegir Columnas Curso, pulsando el botón llamado Exportar Notas del curso. Para la realización del guion llamado ExportCourse, que realiza la exportación, el cuál inicialmente recoge los datos del curso del formulario Elegir Columnas Curso, para luego leer los datos de todas las columnas a partir del siguiente comando de Curl:

curl –k –X GET -H "Authorization: Bearer MvC6nkQApDbjkCNrrMC6CVls57FRY8Zm" http://localhost:9876/learn/api/public/v1/courses/<courseId>/gradebook/columns/?fields=id,name

Donde courseId es recibido como parámetro de la variable local $courseId. Después se recorren todos los valores del portal dentro del formulario Elegir Inscritos, del que se recogen los nombres de las columnas que coinciden con el courseId del id obtenido en la consulta anterior, para concatenarlas en la variable local $columnNames y se finaliza con un salto de línea. Para finalmente leer la nota de cada columna de ese curso, utilizando el siguiente comando Curl:

curl –k –X GET -H "Authorization: Bearer MvC6nkQApDbjkCNrrMC6CVls57FRY8Zm" http://localhost:9876/learn/api/public/v1/courses/<courseId>/gradebook/columns/<columnId>/users/<userId>?fields=score

Con courseId, columnId y userId correspondientes al valor de las variables las locales $courseId, $columnId y $userId obtenidas anteriormente. El resultado es guardado en la variable local $score, la cual puede contener un valor decimal separado por . o contener un valor vacío. El resultado de todas las columnas correspondientes a un alumno concreto, junto con su apellido, nombre y nombre de usuario es concatenado junto con sus notas en una variable local $rows, cada dato se separa con una coma y se finaliza con un salto de línea para adaptarlo al formato csv.

El resultado de las consultas se concatena, como primera fila los nombres de las columnas y después todas sus filas, en el campo text de tabla auxiliar llamada exportar, para luego crear un fichero con el formato csv. El fichero tiene el siguiente nombre "notas_alumnos_<nombre del curso>.csv" . Se ha optado este formato a pesar de que FileMaker permite los siguientes formatos (Valores separados por tabulaciones (*.tab); Valores separados por comas (*.csv); Archivos DBF (*.dbf); Archivos de fusión (*.mer); Archivos de tabla HTML (*.html); Archivos de FileMakerPro (*.fmp 12); Archivos XML (*.xml); Libros de Excel (*.xlsx)) por su sencillez y porque el resultado era una cadena de texto. Al finalizar se abre el fichero automáticamente.

Exportación de las notas de un alumno en un curso concreto

Esta consulta se realiza dentro del formulario Elegir Columnas Alumno, pulsando el botón llamado Exportar Notas del alumno. Para la realización del guion llamado ExportColumn, que realiza la exportación, de forma similar a como lo hace para un curso, pero se prescinde de las subconsultas, al poder obtener los datos al recorrer los portales de los formularios Elegir inscritos y Elegir Columnas Alumno.

Al igual que para la exportación de notas de los alumnos de un curso, se guarda en la misma tabla local auxiliar y se obtiene una concatenación en el campo text que luego es guardado en un fichero csv. El fichero tiene el siguiente nombre "nota_<nombre del alumno>.csv" Al finalizar se abre el fichero automáticamente.

Reuniones para seguimiento y control

Se han realizado diversas reuniones con los tutores para realizar tareas de asesoramiento, seguimiento y control. A continuación, se enumera una breve descripción de cada una de ellas.

Reunión 1

Se trataron los requisitos que necesitaba el proyecto, se suministró una máquina virtual de Blackboard Learn para poder realizar pruebas en un entorno seguro de desarrollo e información asociada de consulta

para poder entender el entorno. También se trató la diferencia entre crear documentación para el cliente y un TFG

Fecha	**1 de Febrero de 2018 a las 12:00**
Asistentes	José Manuel Sota Eguizábal y Sergio Cárcamo García
Duración	35 min.
Lugar	Dpto. de Sistemas - e. Learning - Fundación Universidad de La Rioja

Reunión 2

Se trataron los plazos de entrega, depósito y presentación del TFG. Se habló de qué es un TFG y cómo debería realizarse. Se realizó una idea general de cómo debía planificarse el proyecto y cuáles eran los roles de cada tutor en el mismo, remarcando la idea de que el alumno trabaja solo y que los tutores no son parte del equipo de desarrollo

Fecha	**5 de Febrero de 2018 a las 17:00**
Asistentes	Ángel Luis Rubio García y Sergio Cárcamo García
Duración	20 min.
Lugar	Despacho 3222 - Centro Científico Tecnológico

Reunión 3

Se trataron los problemas encontrados con la configuración de la máquina virtual de Blackboard Learn y la solución aplicada. Se sugirió documentación de consulta sobre el Grade Journey. También se trató la necesidad de seguridad en la aplicación final, que lo ideal es que distinguiese entre los roles Administrador y Profesor, para que la aplicación sepa quién la usa, pero que, si no es posible, que sea usable por un Administrador. Se discutió sobre si utilizar servicios Web Soap para consumir REST y que sería recomendable usar un cliente que consuma REST para no tener que desarrollar nada adicional. Se habló de la importancia de las reuniones y si era necesaria la presencia de todos los tutores

Fecha	**12 de Febrero de 2018 a las 12:00**
Asistentes	José Manuel Sota Eguizábal y Sergio Cárcamo García
Duración	20 min.
Lugar	Dpto. de Sistemas - e. Learning - Fundación Universidad de La Rioja

Reunión 4

Se revisó el Documento de objetivos del trabajo, haciendo hincapié en la importancia de la corrección en la ortografía, una mejora de la introducción para dejar claro cuál es el problema a resolver, la necesidad de dar indicaciones de porqué se ha decidido usar FileMaker en vez de usar otras alternativas, mejorar la descripción de la metodología ágil mediante Sprints y porque se han elegido 30 horas por cada Sprint

Fecha	**12 de Febrero de 2018 a las 17:00**
Asistentes	Ángel Luis Rubio García y Sergio Cárcamo García
Duración	20 min.
Lugar	Despacho 3222 - Centro Científico Tecnológico

Reunión 5

Se revisó el Documento Desarrollo de Servicios REST, que incluía un borrador de los diseños de la aplicación. Se habló de mejoras en el diagrama que mostraba la aplicación en 3 capas. La supresión de un caso de uso de Login por parecer redundante. La adaptación del diagrama de actividad a 2 hojas, ya que no era perfectamente visible. Y la mejora del diagrama de clases, ya que estaba centrado en los objetos del API de Blackboard Learn y no dejaba claro cuales pertenecían a la aplicación

Fecha	**9 de Marzo de 2018 a las 13:00**
Asistentes	Ángel Luis Rubio García y Sergio Cárcamo García
Duración	30 min.
Lugar	Decanato - Centro Científico Tecnológico

Reunión 6

Se trató de revisar los diseños del Documento Desarrollo de Servicios REST con el cliente para concretar si se ajustaban a lo que requería. Se llegó a la conclusión de que usar un API REST escrita en Java y comunicarla con FileMaker era redundante, usar la que incorpora Blackboard Learn es suficiente. Se encontraron varios fallos en el diseño de 3 capas que no concordaban con la documentación de Blackboard. Se trató la posibilidad de tratar de forma más segura las URLs, llegando a la conclusión de que usando OAuth 2 que aporta Blackboard era suficiente. Se trató el tema de quien podría usar la aplicación, el cliente sugirió que se consiguiera primero un acceso total para el administrador, ya que Blackboard es lo que permite por defecto y si sólo si era posible, también para un usuario profesor, con permisos sobre los cursos que le pertenezcan, pero que los usuarios alumno no deberían tener ningún permiso, sólo pertenecer a un curso y poder obtener sus notas

Fecha	**12 de Marzo de 2018 a las 12:00**
Asistentes	José Manuel Sota Eguizábal y Sergio Cárcamo García
Duración	35 min.
Lugar	Dpto. de Sistemas - e. Learning - Fundación Universidad de La Rioja

Reunión 7

Se trataron los resultados obtenidos al usar Curl contra el Api REST de Blackboard. Se concluyó que un sistema de login mediante usuario y contraseña no era posible, pero sí un tratamiento mediante permisos desde FileMaker. Se trató la importancia de cuando realizar una sincronización de la información, si cada vez que había un cambio, cuando el usuario lo decidiese o cuando se fuera a cerrar la aplicación. Se decidió incluir un botón de sincronizar dentro de la interfaz de usuario de FileMaker, para que el usuario decidiese cuando guardar sus cambios y se refrescase la información en pantalla. También se decidió que, si se pulsaba el botón salir, se informase al usuario si quiere sincronizar la información antes de salir o salir sin sincronizar, descartando los cambios. También se habló de la realización de una nueva reunión, con fecha por decidir, con una versión de prueba de la aplicación, en una de las aulas de informática

Fecha	**24 de Abril de 2018 a las 17:00**
Asistentes	Ángel Luis Rubio García, José Manuel Sota Eguizábal y Sergio Cárcamo García
Duración	55 min.
Lugar	Despacho 3222 - Centro Científico Tecnológico

Reunión 8

Se hizo una demostración en una de las aulas de informática de la primera versión estable de la aplicación. Se trataron varios errores en el diseño de la presentación, información redundante y usabilidad de la misma. Se decidió la eliminación de esa información redundante y poco útil para un profesor. La reducción del número de varias pantallas para unificar información útil en una única pantalla. Se sugirió la revisión de la base de datos con la que trabaja FileMaker para adaptarse a esos cambios. Se decidió la eliminación de la opción en el menú de cursos, Inscribir alumno a un curso, ya que generaba mucho tráfico al buscar todos los alumnos que no estaban inscritos en ese curso y era una operación excepcional que sólo podía realizar un profesor

Fecha	**4 de Junio de 2018 a las 11:30**
Asistentes	Ángel Luis Rubio García, José Manuel Sota Eguizábal y Sergio Cárcamo García
Duración	30 min.
Lugar	Aula L130 - Centro Científico Tecnológico

Reunión 9

Se trató la corrección de la memoria y se hizo balance de los resultados obtenidos. La conclusión fue que el estado de la aplicación y la memoria no estaba lo suficientemente avanzado para su presentación en la convocatoria de Junio, por lo que era necesario revisar la aplicación y la memoria.

Fecha	**20 de Junio de 2018 a las 18:00**
Asistentes	Ángel Luis Rubio García y Sergio Cárcamo García
Duración	10 min.
Lugar	Despacho 3222 - Centro Científico Tecnológico

Reunión 10

Se hizo una demostración en una de las aulas de informática de la aplicación revisada. Se trató una revisión de los requisitos y como tratar el rol de administrador. Se habló por primera vez que los formularios de FileMaker mostrados al usuario debían ser similares a como la aplicación Web de Blackboard los muestra, pudiendo filtrar las consultas de forma similar, mediante opciones desplegables, pudiendo introducir un texto y procesándolo con un botón buscar. También la opción de poder discriminar por fechas de creación. Además de poder conseguir la implementación de la exportación de la información a fichero y desde qué formulario sería más óptimo obtenerla.

Fecha	**21 de Junio de 2018 a las 10:00**
Asistentes	Ángel Luis Rubio García, José Manuel Sota Eguizábal y Sergio Cárcamo García
Duración	40 min.
Lugar	Aula L130 - Centro Científico Tecnológico

Reunión 11

Se hizo una demostración en una de las aulas de informática de la aplicación revisada. Se trató cómo aplicar seguridad a nivel de fichero en FileMaker. Se detectó imprecisión en los mensajes de error. Se habló de añadir etiquetas descriptivas a los formularios. Se detectaron errores al sincronizar. Se trató qué hacer con el botón eliminar columnas, si eliminaba sólo la nota del alumno, entonces era admisible, si eliminaba la columna, el botón debía ser eliminado. Se decidió que, en el formulario de alumnos inscritos, sólo debía mostrar los estudiantes. Se detectó que era posible poner notas a un profesor, situación que no debería producirse. Se sugirió revisar el modo de exportación para devolver los resultados texto y usando SQL contra las tablas de FileMaker.

Fecha	**18 de Julio de 2018 a las 10:00**
Asistentes	Aula L130 - Centro Científico Tecnológico
Duración	40 min.
Lugar	Aula L130 - Centro Científico Tecnológico

Seguimiento y control

Estado real de la planificación

N.º de Tarea	Nombre de la Tarea	Duración en horas	Inicio	Fin	Predecesor	Estado
G2.1	Reunión con el Tutor de la Empresa	0.5833	Jueves 01/02/18	Jueves 01/02/18		Finalizado
S1	Configuración de la Plataforma Blackboard Learn	28	Jueves 01/02/18	Lunes 12/02/18		Finalizado
G2.2	Reunión con el Tutor de la Universidad	0.3333	Lunes 05/02/18	Lunes 05/02/18		Finalizado
G1.1	Planificación	20	Lunes 05/02/18	Viernes 09/02/18		Finalizado
G1.2	Documento de objetivos del trabajo	20	Lunes 05/02/18	Lunes 12/02/18		Finalizado
G2.3	Reunión con el Tutor de la Universidad	0.3333	Lunes 12/02/18	Lunes 12/02/18	G1.2	Finalizado
G2.4	Reunión con el Tutor de la Empresa	0.3333	Lunes 12/02/18	Lunes 12/02/18	S1	Finalizado
S2	Estudio del panel de administración Blackboard Learn	28	Martes 13/02/18	Jueves 22/02/18	S1	Finalizado
S3	Desarrollo Servicios REST	32	Lunes 26/02/18	Jueves 08/03/18	S2	Finalizado
G2.5	Reunión con el Tutor de la Universidad	0.5	Viernes 09/03/18	Viernes 09/03/18	S3	Finalizado
G2.6	Reunión con el Tutor de la Empresa	0.5833	Viernes 09/03/18	Viernes 09/03/18	S3	Modificada la fecha al lunes 12/03/18 Finalizado
S4	Implementación del Middleware	0	Martes 13/03/18	Viernes 23/03/18	S3	Descartado por no usar implementación en Java
G3.1	Pruebas de los Servicios REST	66	Lunes 26/03/18	Martes 27/03/18	S3	Adelantado al martes 13/03/18 Finalizado el Viernes 20/04/18
G2.7	Reunión con el Tutor de la Universidad	0.4583	Lunes 28/03/18	Lunes 28/03/18	S4; G3.1	Modificada la fecha a martes 24/04/18

						Finalizado
G2.8	Reunión con el Tutor de la Empresa	0.4583	Lunes 28/03/18	Lunes 28/03/18	S4; G3.1	Modificada la fecha a martes 24/04/18 Finalizado
S5	Diseño de los formularios en FileMaker	44	Jueves 29/03/18	Martes 10/04/18	S4; G3.1	Retrasado Inicio a martes 24/04/18 Finalizado el Miércoles 13/06/18
G2.9	Reunión con el Tutor de la Universidad	0.25	Lunes 11/04/18	Lunes 11/04/18	S5	Modificada la fecha a lunes 04/06/18 Finalizada
G2.10	Reunión con el Tutor de la Empresa	0.25	Lunes 11/04/18	Lunes 11/04/18	S5	Modificada la fecha a lunes 04/06/18 Finalizada
S6	Implementación de la funcionalidad en FileMaker	39	Jueves 12/04/18	Lunes 23/04/18	S5	Retrasado Inicio a martes 24/04/18 Finalizado el Miércoles 13/06/18
G3.2	Pruebas de la funcionalidad en FileMaker	36	Martes 24/04/18	Jueves 26/04/18	S6	Retrasado Inicio a martes 24/04/18 Finalizado el Miércoles 13/06/18
G2.11	Reunión con el Tutor de la Universidad	0	Viernes 27/04/18	Viernes 27/04/18	S6; G3.2	No se ha utilizado
G2.12	Reunión con el Tutor de la Empresa	0	Viernes 27/04/18	Viernes 27/04/18	S6; G3.2	No se ha utilizado
S7	Memoria	24	Lunes 30/04/18	Martes 08/05/18	S1; G1.1; G1.2; S2; S3; S4; G3.1; S5; S6; G3.2	Finalizado el Viernes 15/06/18
S8	Presentación	0	Miércoles 09/05/18	Viernes 18/05/18	S7	No se ha utilizado
S9	Ensayos	0	Lunes 21/05/18	Lunes 21/05/18	S8	No se ha utilizado
G2.13	Reunión con el Tutor de la Universidad	0	Viernes 08/06/18	Viernes 08/06/18	S7; S8; S9	No se ha utilizado

	Reunión con el Tutor de la Empresa	0.1666	Miércoles 20/06/18	Miércoles 20/06/18		Finalizado
	Ampliación del Plazo a Julio					
	Reunión con el Tutor de la Universidad	0.3333	Jueves 21/06/18	Jueves 21/06/18		Finalizado
	Reunión con el Tutor de la Empresa	0.3333	Jueves 21/06/18	Jueves 21/06/18		Finalizado
	Planificación	27	Viernes 22/06/18	Lunes 27/06/18		Finalizado
	Implementación de la funcionalidad en FileMaker	30	Martes 28/06/18	Lunes 09/07/18		Finalizado
	Pruebas de la funcionalidad en FileMaker	14	Martes 10/07/18	Viernes 13/07/18		Finalizado
	Memoria	14	Martes 10/07/18	Viernes 13/07/18		Finalizado
	Reunión con el Tutor de la Universidad	0.3333	Miércoles 18/07/18	Miércoles 18/07/18		Finalizado
	Reunión con el Tutor de la Empresa	0.3333	Miércoles 18/07/18	Miércoles 18/07/18		Finalizado
	Presentación	0	Sin asignar	Sin asignar		Pendiente
	Ensayos	0	Sin asignar	Sin asignar		Pendiente
	Total	427.6099				

Listado de Modificaciones

- Tras la reunión con el tutor de la empresa, se eliminaron los servicios Web Soap del Documento de objetivos del trabajo y de la planificación.
- Tras la reunión con el tutor, se corrigió el Documento de objetivos del trabajo.
- Por mail se comunicó a los tutores que se estaba realizando una traducción del inglés al castellano de la documentación del API REST de Blackboard Learn. Se decidió abandonar esa documentación adicional, porque iba a consumir demasiado tiempo.
- Tras la reunión con el tutor, se modificó el diagrama de clases en su versión para Java, para distinguir mejor cuales eran los objetos de la aplicación, en esa versión estaban más centrados en el API REST de Blackboard Learn. También se modificó el diagrama de la aplicación para incorporar unas pequeñas mejoras. Se eliminó un caso de uso de Login por parecer redundante.
- Tras la reunión con el tutor de la empresa se decidió abandonar la implementación middleware del API REST en Java por ser redundante. Este cambio de dirección llevó a la modificación del diagrama de objetos para adaptarse a cómo trata los datos FileMaker. Se corrigió el diseño de la aplicación en 3 capas porque no concordaba con la documentación de Blackboard. Se modificaron los casos de uso para adaptarse mejor a la idea de usuario profesor y superusuario administrador.
- Después de la finalización del Sprint 3, ha habido retrasos y el resto de Sprints no han podido ser cumplidos. Se han ido dando prioridad a las tareas según iban apareciendo nuevas modificaciones y se han ido corrigiendo los errores que surgían.
- Tras la reunión con los tutores se incluyó en la aplicación un tratamiento de permisos desde FileMaker que sustituía el caso de uso de Login. Se añadió a la aplicación el concepto de sincronizar mediante un botón.
- Tras la reunión de demostración de la aplicación con los tutores, se simplificó la información mostrada en pantalla, eliminando aquella que se dijo ser redundante para un profesor. Se redujo

el número de pantallas, ya que había una relación entre tablas y presentaciones. Se modificó el diseño de clases, orientado inicialmente más al formato de base de datos relacional, a un formato de base de datos no relacional, ya que FileMaker no consumía bien los datos. Se eliminó el caso de uso Inscribir alumno a un curso, ya que generaba mucho tráfico y era una operación excepcional que sólo podía realizar un profesor.

- Se han eliminado de los casos de uso todas las operaciones de modificación de usuarios.
- Se ha excluido del proyecto la funcionalidad de poder exportar los resultados a otro formato por falta de tiempo.
- Se ha modificado la planificación inicial para adaptarse a la ampliación de plazo para finales de Julio, en la memoria se ha mantenido esa planificación inicial, pero aparecen los cambios reales, que aparecen como “Ampliación del Plazo a Julio”.
- Se han modificado la mayoría de los casos de uso para adaptarse a los nuevos requisitos y se ha añadido un diagrama de actividad individual por cada uno de los distintos Formularios que componen la aplicación.
- Se ha revisado el diagrama de clases para adaptarse a los nuevos requisitos.
- Se han adaptado los formularios para hacerlos similares a como la aplicación Web de Blackboard los muestra, pudiendo filtrar las consultas de forma similar, mediante opciones desplegables, pudiendo introducir un texto y procesándolo con un botón buscar. También poder discriminar por fechas de creación.
- Se ha recuperado la posibilidad de exportar la información a fichero, pudiendo realizarla por cursos (para cada alumno, exportando individualmente cada columna de notas) o por alumnos (para cada columna de notas de forma individual).
- Se ha añadido una contraseña a nivel de fichero para el usuario Admin
- Se han revisado los mensajes de error mostrados al usuario para hacerlos más precisos
- Se han añadido etiquetas de título a los formularios
- Se han modificado las exportaciones para que las columnas de notas estén agrupadas por alumno
- Se ha revisado la sincronización y la actualización de los portales relacionados tras la misma
- Se ha cambiado el nombre del botón Cerrar sesión, en todas sus apariciones por Menú Inicial
- Al buscar por curso, en el filtro, ahora, por defecto se muestra Buscar por curso.
- En el formulario de alumnos inscritos, ahora sólo muestra los usuarios que sean estudiantes.
- Se ha añadido una autorelación entre los cursos y las columnas de notas
- Se ha añadido un formulario para hacer modificaciones en las columnas de un curso
- Se ha eliminado el botón eliminar porque eliminaba permanentemente las columnas

Conclusiones

El objetivo prioritario del proyecto se ha cumplido, que era permitir a un profesor o a un administrador del sistema, poder modificar fácilmente las notas discriminando por cursos, los alumnos que pertenecen a esos cursos y por último por un sistema de columnas que puede gestionar el usuario. Esta funcionalidad ya existía en la plataforma web, pero no existía nada parecido en la plataforma sugerida por el tutor de la empresa, FileMaker, mi aplicación, aunque sencilla, es funcional. En un futuro podría mejorarse fácilmente, ya que el sistema de guiones de FileMaker usando la función Insertar desde URL, permite recibir información adicional en Json que yo no he incluido, como por ejemplo los grupos de alumnos que pertenecen a un curso, los permisos del sistema, el sistema de preguntas de evaluación y que puede ser accedida mediante el API Rest de Blackboard; mostrando esa información en presentaciones específicas de la tarea que necesitemos.

El objetivo secundario del proyecto, que era la exportación de la información referente a las notas (cada columna tiene asignada una nota) de los alumnos de un determinado curso, también ha sido cumplido. Adicionalmente, también es posible realizar la exportación de las notas (corresponden sólo a las columnas de notas de ese alumno en el curso activo) de un alumno concreto.

Se ha utilizado la función de guion Exportar registros, que permite al usuario exportar a un archivo local (Valores separados por tabulaciones (*.tab); Valores separados por comas (*.csv); Archivos DBF (*.dbf); Archivos de fusión (*.mer); Archivos de tabla HTML (*.html); Archivos de FileMakerPro (*.fmp 12); Archivos XML (*.xml); Libros de Excel (*.xlsx)), sin embargo, se ha elegido el formato csv porque los resultados eran devueltos en una cadena, con la primera fila los nombres de las columnas, separadas por comas y cada fila separada por un salto de línea.

El 90% del proyecto ha sido una tarea de investigación, ya que en la carrera apenas se había tocado la API Rest (aparecía como una práctica, que además era opcional en la asignatura Sistemas Distribuidos, utilizando el lenguaje Java y el API Jersey), la plataforma FileMaker la había usado superficialmente (en las prácticas de empresa la utilicé con un enlace ODBC contra mysql para usar una base de datos externa y sin usar la base de datos nativa de la plataforma, los guiones usados sólo se ocupaban de validar URLs o de saltar de un menú a otro sin mayor complejidad) y en esta ocasión tuve que crear desde cero una base de datos no relacional, nativa de FileMaker (en la carrera todas las asignaturas relacionadas con bases de datos usaban como base sistemas gestores de bases de datos relacionales y SQL para realizar las consultas). Blackboard sólo lo había usado en su formato web y como usuario, no sabía nada de cómo funcionaba internamente. Tampoco había usado nunca Curl y no conocía su existencia, al buscar información para consumir los datos del API, di con una referencia a él, en los foros de Blackboard. Tal situación ha provocado que ante mi mala formación (los conocimientos adquiridos en la carrera eran insuficientes, no por falta de ellos, sino por nula utilidad en la tarea encomendada) que el hacer una planificación inicial fracasara y tuviera que replanificar en varias ocasiones para poder adaptar el proyecto a los nuevos requisitos.

El 10 % del resto del proyecto ha sido poner en práctica lo aprendido y resolver los problemas que surgían, con un alto grado de improvisación, ya que los conocimientos previos a la investigación, no eran útiles.

La planificación mediante Sprints ha fracasado completamente, ya que no ha habido una retroalimentación real de información entre el alumno (el desarrollador) y el tutor de la empresa (el cliente). Después de la ampliación del plazo y sin la utilización de Sprints, ni ningún tipo de plazos intermedios, parece haber mejorado la retroalimentación, ya que el proyecto por fin parece estar más al gusto de todos.

Bibliografía

Relacionado con Virtualización:

- Descarga de VirtualBox
- Descarga de Vagrant
- Consulta de modelo de procesador Intel

Relacionado con el campus Virtual:

- Página para desarrolladores de Blackboard
- Noticias relacionadas con programación para Blackboard Learn

Relacionado con Java:

- Descarga del JVM
- Descarga de Tomcat
- Descarga de OXYGEN
- Descarga de Jersey
- Ejemplo de implementación del API REST en Java
- Ejemplo de autenticado con OAuth 2 en Java

- Guía de drivers JDBC para FileMaker
- Tutorial para crear un servicio Rest con Eclipse

Relacionado con el API REST:

- Documentación del API REST de Blackboard
- Descarga de CURL
- Códigos de estado HTML
- Ejemplos de uso del API REST mediante CURL

Relacionado con FileMaker:

- Ayuda de FileMaker 16